高等学校计算机专业教材精选·网络与通信技术

Web开发技术实验指导

陈轶 主编　　张薇 肖建 王炜立 副主编



清华大学出版社

高等学校计算机专业教材精选·网络与通信技术

Web开发技术实验指导

陈　铁　主编
张　薇　肖　建　王炜立　副主编

清 华 大 学 出 版 社
北　京

内 容 简 介

本书是《Web开发技术实用教程》(ISBN 978-7-302-17435-6)的配套实验指导教材。本书分成14章,内容安排合理,实用性强。涵盖了当前Web开发技术的主要内容,具体涉及常见的开发工具Dreamweaver CS 3.0和SharePoint Designer 2007、HTML技术基础、XHTML技术、CSS技术、客户端脚本语言、JSP开发的Java语言基础、JSP的开发体系和环境配置、JSP的主要内置对象、JSP的其他内置对象、JSP的文件操作、JSP的JavaBean编程、JSP的Servlet编程、JSP访问Web数据库、XML技术以及Web的综合应用。并介绍了开发无线WAP 2.0的标记语言和脚本语言,利用它们开发无线Web应用。

此外,对Web开发中常见的Java API,如DOM4J、JDOM、jspSmartUpload组件、JavaMail API等都有所涉及,使读者开发Web的高级应用成为可能。另外,在每一章都提供了各种形式具有现实意义的实验练习,帮助读者了解和掌握Web相关技术,为进一步开发Web应用提供保证。

本书可以作为高等学校计算机及相关专业学生的Web程序设计、Web技术、网页设计、JSP技术课程教材,也可供技术人员使用。

图书在版编目(CIP)数据

Web开发技术实验指导/陈轶主编. —北京:清华大学出版社,2009.8
(高等学校计算机专业教材精选·网络与通信技术)
ISBN 978-7-302-19942-7

Ⅰ. W…　Ⅱ. 陈…　Ⅲ. 主页制作—程序设计—高等学校—教学参考资料　Ⅳ. TP393.O92

中国版本图书馆CIP数据核字(2009)第059230号

责任编辑: 汪汉友
责任校对: 焦丽丽
责任印制:

出版发行: 清华大学出版社　　**地　址:** 北京清华大学学研大厦A座
http://www.tup.com.cn　　**邮　编:** 100084
社 总 机: 010-62770175　　**邮　购:** 010-62786544
投稿与读者服务: 010-62776969,c-service@tup.tsinghua.edu.cn
质 量 反 馈: 010-62772015,zhiliang@tup.tsinghua.edu.cn
印 刷 者:
装 订 者:
经　销: 全国新华书店
开　本: 185×260　**印　张:** 18.25　**字　数:** 445千字
版　次: 2009年8月第1版　**印　次:** 2009年8月第1次印刷
印　数: 1～0000
定　价: 0.00元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题,请与清华大学出版社出版部联系调换。
联系电话:010-62770177转3103　　产品编号:030059-01

出 版 说 明

我国高等学校计算机教育近年来迅猛发展，应用所学计算机知识解决实际问题，已经成为当代大学生的必备能力。

时代的进步与社会的发展对高等学校计算机教育的质量提出了更高、更新的要求。现在，很多高等学校都在积极探索符合自身特点的教学模式，涌现出一大批非常优秀的精品课程。

为了适应社会的需求，满足计算机教育的发展需要，清华大学出版社在进行了大量调查研究的基础上，组织编写了《高等学校计算机专业教材精选》。本套教材从全国各高校的优秀计算机教材中精挑细选了一批很有代表性且特色鲜明的计算机精品教材，把作者们对各自所授计算机课程的独特理解和先进经验推荐给全国师生。

本系列教材特点如下。

(1) 编写目的明确。本套教材主要面向广大高校的计算机专业学生，使学生通过本套教材，学习计算机科学与技术方面的基本理论和基本知识，接受应用计算机解决实际问题的基本训练。

(2) 注重编写理念。本套教材作者群为各校相应课程的主讲，有一定经验积累，且编写思路清晰，有独特的教学思路和指导思想，其教学经验具有推广价值。本套教材中不乏各类精品课配套教材，并力图努力把不同学校的教学特点反映到每本教材中。

(3) 理论知识与实践相结合。本套教材贯彻从实践中来到实践中去的原则，书中的许多必须掌握的理论都将结合实例来讲，同时注重培养学生分析、解决问题的能力，满足社会用人要求。

(4) 易教易用，合理适当。本套教材编写时注意结合教学实际的课时数，把握教材的篇幅。同时，对一些知识点按教育部教学指导委员会的最新精神进行合理取舍与难易控制。

(5) 注重教材的立体化配套。大多数教材都将配套教师用课件、习题及其解答，学生上机实验指导、教学网站等辅助教学资源，方便教学。

随着本套教材陆续出版，相信能够得到广大读者的认可和支持，为我国计算机教材建设及计算机教学水平的提高，为计算机教育事业的发展做出应有的贡献。

清华大学出版社

前　言

伴随着 Internet 应用领域的不断扩展和 Web 技术的突飞猛进，"Web 开发技术"以及相关课程已经得到许多学校的关注。为了适应计算机教学发展趋势以及主流 Web 技术发展现况，有必要编写一本符合当前 Web 技术发展趋势和教学现状的实验教材，帮助广大学生了解和掌握 Web 的主流技术。

本书是清华大学出版社出版《Web 开发技术实用教程》(ISBN 978-7-302-17435-6)的配套实验教材，是作者总结多年 Web 项目开发以及 Web 教学实践的经验，结合相关的技术资料编写而成。本书从程序设计角度出发，紧密结合 Web 开发技术的特点和高校 Web 开发课程的教学大纲，力求覆盖当前主流 Web 开发中使用的常用技术，涉及了当前 Web 应用中常见的客户端、服务器端技术，侧重介绍了服务器端编程常用 JSP 技术的基本内容。此外，本书还引入了 WAP 2.0 编程，引导读者进入无线 Web 开发领域，有效填补当前同类教材在无线应用领域的空白。

本书分成 14 章，涵盖了当前 Web 开发技术中的 Web 技术基础、Web 客户端开发技术和服务器端开发技术 3 部分部分内容，具体如下。

第 1 部分　Web 技术基础

第 1 章，了解 Web 开发技术基础，具体设计了 Web 的基本概念、发展状况、工作原理、工作模式以及关键技术，介绍常见的开发工具 Dreamweaver CS 3.0 和 Microsoft Office SharePoint Designer 2007。

第 2 部分　Web 客户端开发技术

第 2 章 HTML 和 XHTML。从脚本语言发展的角度出发，依次介绍了 HTML 的基本概念以及常见的基本标签、XHTML 主要内容以及设计网站的基本建设要素。并设计了表格、表单、框架结构等多种 XHTML 元素的具体的应用练习，充分展示了 XHTML 的优势。

第 3 章 CSS 技术。介绍了 CSS 基本语法以及常见的基本属性。使读者了解如何用 CSS 创建生动的网页外观。并设计了 CSS 选择符、CSS 结合 DIV 设计菜单以及用 CSS 布局的相关实验练习，让读者对 CSS+DIV 实现网页设计有一个初步的印象。

第 4 章客户端脚本语言。主要介绍 JavaScript 脚本语言，具体涉及的内容有 JavaScript 脚本语言的基本语法、JavaScript 的控制流程、JavaScript 的函数、JavaScript 的事件处理、JavaScript 的内置对象。通过对它们的介绍，理解 CSS+DIV+JavaScript 开发 DHTML，并为进一步学习 AJax 打下坚实的基础。

第 5 章可扩展标记语言 XML。主要介绍并设计了 XML 在客户端的应用实验，具体包括 XML 基础、XML 的验证机制 DTD 和 XML Schema、XML 的 CSS 显示以及 XPATH 基础和 XSLT 转换 XML。为读者后续学习 XML 在服务器端开发打下基础。

第 6 章 WAP 2.0 编程。介绍 WAP 2.0 无线应用协议，具体涉及的内容有 WML 1.3、WML 2.0、XHTML Basic、XHTML Mobile Profile、WMLScript 等。通过它们引导读者了解各无线移动受限设备终端的 WAP 网页设计，无线终端的移动 Web 应用的开发。

第 3 部分　Web 服务器端开发技术

比较了常见的服务器端的常见开发语言，并侧重介绍了服务器端常用的编程语言 JSP。

第 7 章 JSP 开发的 Java 语言基础。这使没有任何编程经验的读者可以迅速进入学习状态而特别编写的。具有 Java 编程经验的读者可以跳过这一章。这一章主要设计了异常处理和多线程处理的实验。

第 8 章 JSP 简介。介绍了 JSP 的工作原理和 JSP 基本语法。引导读者学习安装并配置 Tomcat 服务器，进入学习开发 JSP 应用，并让读者了解开发购物车的基本原理。

第 9 章 JSP 的内置对象。介绍了 JSP 的 out、request、response、application、session、page、pageContext、config 和 exception 等 9 种内置对象，并设计具体实例练习强化对这些内置对象的理解。

第 10 章 JSP 的文件操作。介绍了 JSP 实现对文件的操作，具体内容有 File 类、JSP 的输入流和输出流、文件的相关操作，如文件的写入、读取、修改、目录的访问以及文件的上传等与文件的下载。

第 11 章 JSP 访问 Web 数据库。这一章是 Web 开发的一个重要内容。介绍了 JDBC、JSP 访问数据的相关操作以及实现、JSP 访问数据库的常见技巧，如中文字符乱码问题的解决、分页显示、连接池的使用等内容。并设计了职员管理系统开发练习，了解 JSP 访问 Web 数据库。

第 12 章 JSP 的 JavaBean 编程。介绍了 JavaBean 组件技术、JavaBean 访问数据库以及 JSP 中的使用 JavaBean 开发具体的 JSP 应用。并介绍了 JavaMail API，通过实验练习让读者了解开发 Web Mail 系统的基本原理。

第 13 章 JSP 的 Servlet 编程。介绍 Servlet 技术，设计相关的练习帮助读者了解 Servlet 与 JavaBean 和 JSP 技术结合的开发模式。并针对 Servlet 实现会话管理、Servlet 实现文件管理、Servlet 实现数据库操作和 Servlet 绘制图形相关知识点，并设计了相应的练习。

第 14 章 JSP 和 XML。主要介绍 JSP 是如何实现 XML 在服务器端的应用。是第 5 章的深入。具体介绍的内容有 JSP 生成 XML、JSP 应用 DOM API、SAX 2.0 API、DOM4J 和 JDOM 解析 XML、JSP 应用 XML，并对 JSP 的自定义标签展开说明。

为了方便读者练习，本书提供了多媒体教案以及教材介绍的实例的源代码，均可在清华大学出版社网站上下载。

本书由南昌大学的陈铁主持编写，南昌大学的王炜立、肖建、李文、邱桃荣、姚力文、姚晓昆、唐祎玲、华东交通大学的张薇和江西省计算中心的杨宇仙参与编写。其中本书第 6 章、第 12 章、第 13 章和第 14 章由陈铁编写，第 5 章与第 9 章由陈铁和张薇合写，第 2 章由陈铁和杨宇仙合写，第 1、3、4、8 章由肖建编写。第 10、11 章由王炜立编写，第 7 章由李文编写，实验答案由姚晓昆整理。最后由陈铁统稿。邱桃荣和姚力文两位老师是本书的特别技术指导，对本书的编写起到重要作用。江西省计算中心的杨国强研究员和陈征研究员对全书进行审阅，并提供了许多宝贵建议。另外，在本书的编写过程中得到了南昌大学的王命延教授等各位老师的大力协助，在此表示衷心的感谢。

由于编者水平所限，书中难免存在错误和不足之处，恳请广大读者对本书的提供宝贵意见和建议。

编　　者

目　录

第 1 章　Web 技术

Web 全称为 World Wide Web(简称 WWW,也就是人们所熟悉的万维网),是 Internet 提供的一种信息服务。Web 汇集了各种不同类型的信息,它的页面颜色丰富、包含文字、图形、动画、声音和视频等多种信息内容,随着 Web 开发技术的不断发展,几乎所有的信息技术领域都或多或少受到 Web 的影响。

1.1　预备知识

1.1.1　Internet 相关概念

1. IP 地址

IP 地址是识别 Internet 中的主机及网络设备的唯一标识。每个 IP 地址可以分为网络地址和主机地址两部分,长度为 32 位(4B),由 4 个十进制数通过“.”分隔组成,每个十进制数的取值范围为 0～255,描述形式如:192.168.0.1。

IP 地址可以分为 5 类:A 类地址、B 类地址、C 类地址、D 类地址和 E 类地址。

2. 域名

TCP/IP 协议提供了域名管理系统 DNS(domain name system)来为每个主机分配字符名称,也就是域名,访问网络时该系统会自动实现域名与 IP 地址的转换。Internet 中域名采用分级命名的机制,基本结构如下:

主机名.三级域名.二级域名.顶级域名

3. URL

URL 提供了待访问信息资源在 Internet 上的准确位置,描述了要访问该资源所使用的访问方式、提供 Web 服务的主机名、待访问文档在主机上的路径和文件名,以及客户机与远程主机通信时使用的端口号。它的基本格式如下:

<访问方式>://<主机名>:<端口号>/<文件路径>

1.1.2　Web 技术概述

Web 站点的开发可以分成客户端和服务器端两部分,客户端主要用于显示信息内容,服务器端程序主要是对所需信息进行处理。

常用的 Web 客户端开发技术有 HTML、CSS 和脚本语言等,而常用的 Web 服务器端开发技术有 JSP、ASP、PHP、ASP. NET 和 XML 等。

Web 采用浏览器/服务器(Browser/Server,B/S)工作模式。用户启动客户端浏览器,在浏览器中输入要访问的 URL 地址,由浏览器向 Web 服务器发出请求;服务器根据客户端发送的请求找到相应文件,如文件是 HTML 文档,则将该文档直接返回给客户端,如果文

件中包含 JSP、ASP 或 PHP 程序，则由 Web 服务器运行该程序并把运行结果返回给客户端，如果程序中包含对数据库的操作，则服务器将指令发送给数据库驱动程序，由数据库驱动程序执行相关指令并将执行结果返回给 Web 服务器，然后再通过服务器将数据运行结果嵌入页面并将完整的 HTML 页面返回给客户端浏览器；客户端接收到返回页面后通过浏览器将返回结果呈献给用户。

1.1.3 HTTP 概述

HTTP 是用于从 WWW 服务器传送文件到本地客户端浏览器的传送协议，它是基于请求/响应的工作模式，当 Web 浏览器和服务器间用 HTTP 协议来传送文档时，它的工作过程可以理解如下：

(1) 在浏览器地址栏输入 URL 地址后，由浏览器向 DNS 请求解析该 URL 对应的 IP 地址，并向该 IP 地址对应的服务器发送建立连接的请求。

(2) 浏览器与服务器建立 TCP 连接。

(3) 服务器给出响应，将被访问文件发回给浏览器。

(4) TCP 连接被释放。

(5) 客户端和服务器断开连接。

1.1.4 Dreamweaver CS 3.0 简介

Dreamweaver CS 3.0 是由 Adobe 公司推出的一款专业的网页制作工具，它主要用于对 Web 站点、页面和 Web 应用程序进行设计、编码和开发，通过 Dreamweaver CS，用户可以高效地设计开发和维护网站和应用程序。Dreamweaver CS 使用方便，对于没有熟练掌握 HTML 标记和脚本语言的用户，使用它也可以很轻松地制作出 Web 页面。它提供了强大的编辑环境使代码编辑更加方便，并可以支持多种服务器技术，同时还具备后台文件传输功能。

Dreamweaver CS 3.0 的工作界面如图 1-1 所示，包括菜单栏、插入选项栏、文本编辑区、属性面板和面板组等。

(1) 菜单栏。菜单栏包含设计、开发、测试 Web 页面和 Web 应用程序等多个菜单命令。菜单栏主要包括“文件”、“编辑”、“查看”、“插入记录”、“修改”、“文本”、“命令”、“站点”、“窗口”和“帮助”这 10 个菜单。其中当单击“插入记录”选项时，可以在其提供的子菜单中选择需要的 HTML 元素直接添加到当前正在编辑的页面中，其子菜单如图 1-2 所示。

(2) “插入记录”选项栏。“插入记录”选项栏位于 Dreamweaver 工作界面中菜单栏的下方，它以按钮形式提供了一些常用的插入选项按钮，如“超链接”、“电子邮件链接”、“表格”、“DIV 标签”、“图像”、“媒体”和“日期”等。

(3) 文本编辑区。文本编辑区包含了文档工具栏和文本编辑窗口。文本编辑窗口是用户使用最多的地方，可以在文本编辑窗口打开多个文件，但在同一时刻只有一个文件处于活动状态，可以被编辑。文本编辑窗口上方的文档工具栏用于在文本编辑窗口中切换文件的各种视图，可以从文本编辑窗口看到处于活动状态的文件的代码视图和设计视图。

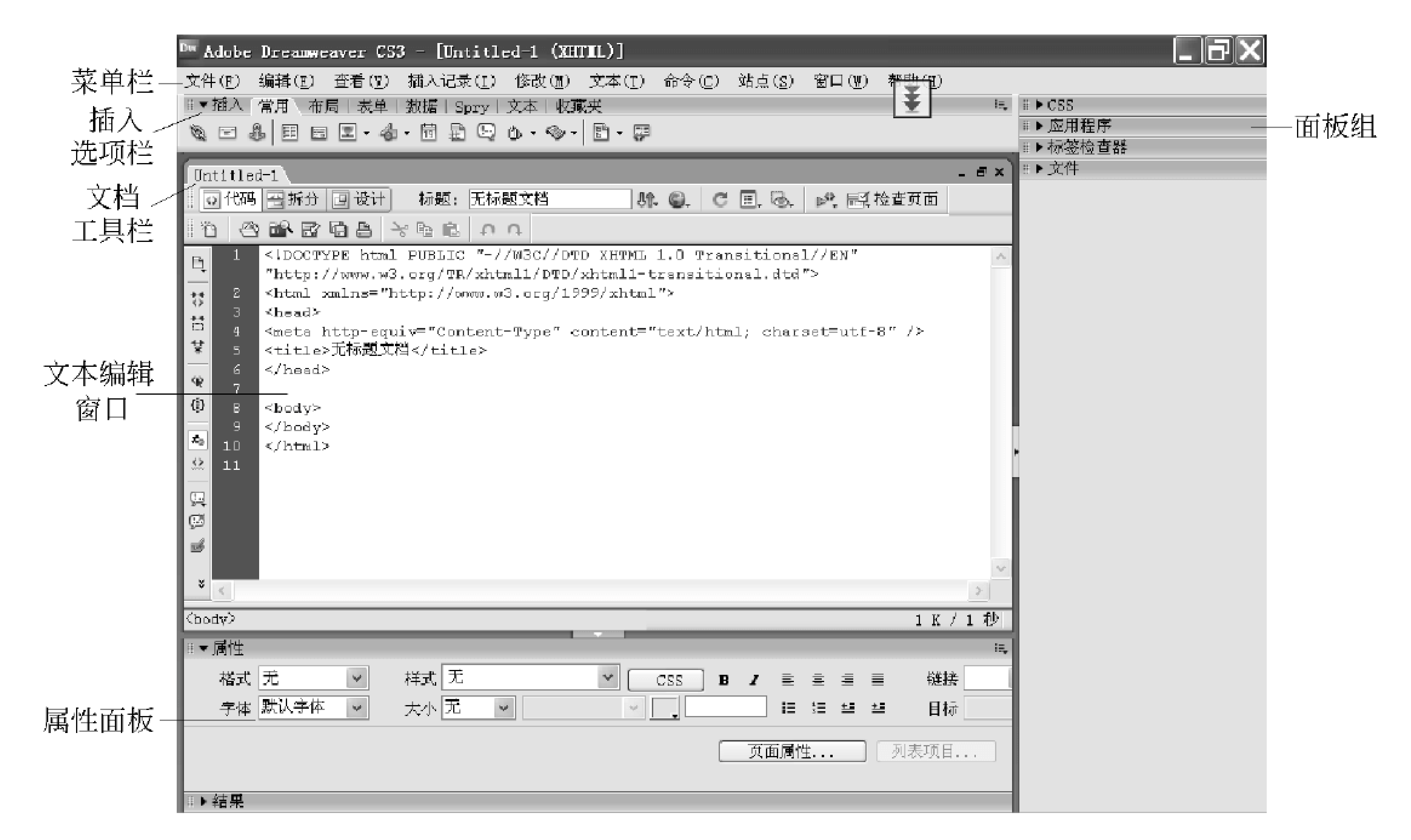


图 1-1　Dreamweaver CS 3.0 的工作界面



图 1-2　“插入记录”选项的子菜单

（4）属性面板。在文本编辑窗口中选择某一个已经插入的页面元素时，在属性面板中会自动出现该元素的相关属性并可以在此设置和修改属性的值。属性面板中显示的内容将随所选定的不同元素而不同。

（5）面板组。面板组中包含了 CSS 样式面板、应用程序面板、标签检查器面板和文件面板等内容，在面板中的操作会直接显示在当前正在编辑的文档中。

1.1.5 Office SharePoint Designer 2007 简介

Office SharePoint Designer 2007 是由 Microsoft 推出的一款新产品，用来基于 SharePoint 技术创建和自定义 SharePoint 网站并生成启用工作流的应用程序，也提供了开发 Web 页面和 Web 应用程序的集成开发环境。Office SharePoint Designer 2007 提供了多种专业工具，利用这些工具，用户在该环境中无须编写代码即可生成交互解决方案、设计自定义网站以及使用报告和托管权限维护网站性能。

Office SharePoint Designer 2007 的工作界面如图 1-3 所示，包括菜单栏、工具栏、文档编辑窗口、文件夹列表、标记属性和工具箱等。

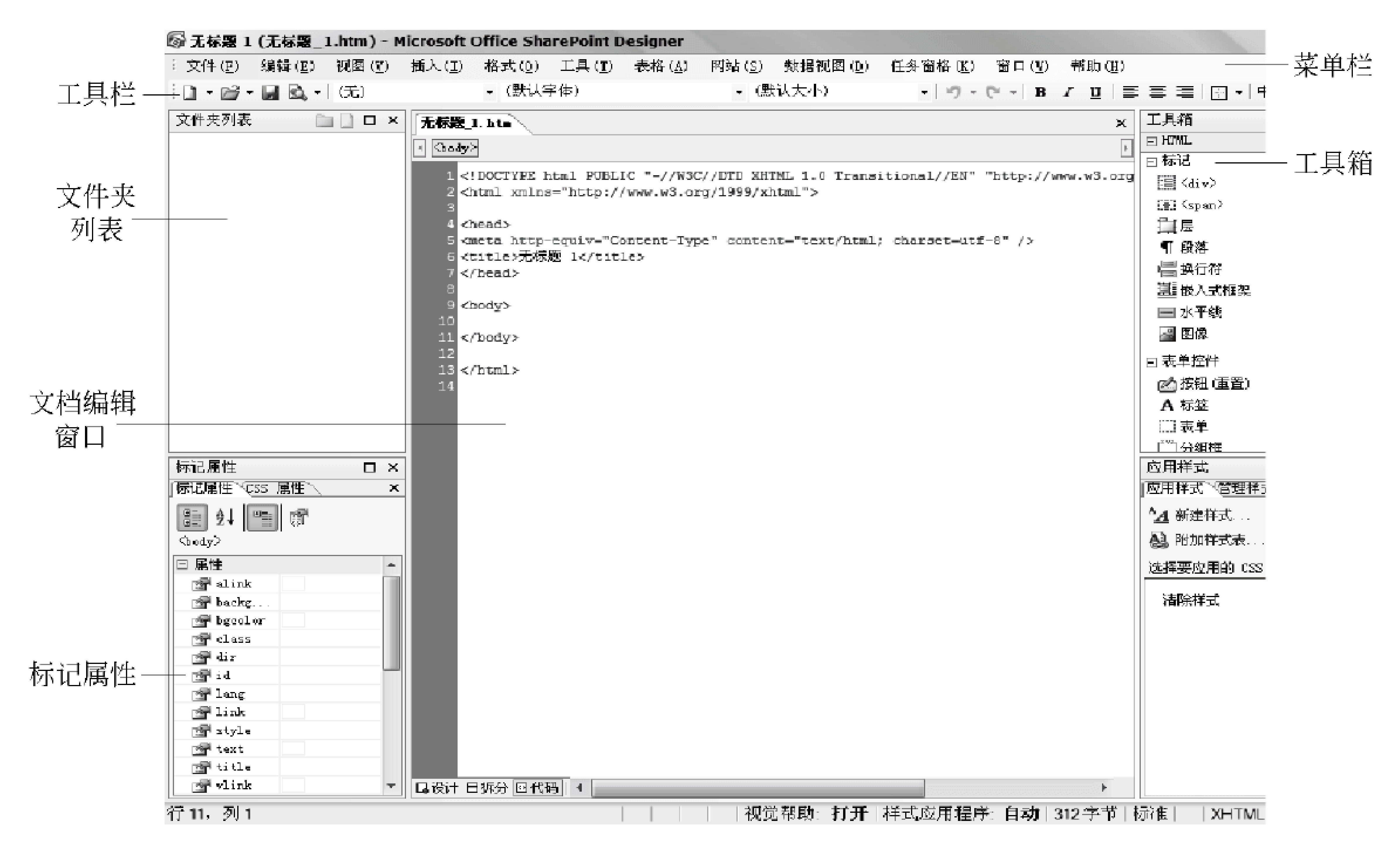


图 1-3 SharePoint Designer 2007 的工作界面

(1) 菜单栏。菜单栏包含设计、开发、编辑 Web 应用程序等多个菜单命令。菜单栏主要包括“文件”、“编辑”、“视图”、“插入”、“格式”、“工具”、“表格”和“网站”等 12 个菜单项。

(2) 工具栏。工具栏位于 SharePoint Designer 工作界面中菜单栏的下方，它以按钮形式提供了一些常用功能按钮，如“创建新文件”、“打开文件”、“保存”和“查询”等。

(3) 文档编辑窗口。我们可以在文档编辑窗口查看多个不同文件的内容，但在同一时刻只有一个文件处于活动状态，可以被编辑。文档编辑窗口下方的文件视图标签用于在文档编辑窗口中切换文件的不同视图，包括文档的源代码和文档的设计视图。

(4) 文件夹列表。在文件夹列表中显示了当前处于活动状态的待编辑文档所在文件夹的结构。

(5) 标记属性。标记属性窗口中列出了在文档编辑窗口中所选定的 HTML 标记可设定的相关属性信息。

（6）工具箱。工具箱中列出了在 Web 程序开发过程中常用的各种工具，如各种 HTML 标记、已定义的样式表等。可以在此选择需要的标记或样式直接插入到待编辑的文档中，方便了程序的开发。

1.2　实验 1.1　使用 Dreamweaver CS 新建站点

实验目的：

（1）熟悉 Dreamweaver CS 的使用。

（2）了解如何使用 Dreamweaver CS 开发页面。

实验内容：

（1）熟悉 Dreamweaver CS 3.0 的开发环境，使用 Dreamweaver CS 3.0 新建一个站点。

（2）用 Dreamweaver CS 3.0 开发一个简易的网页页面。

实验步骤：

练习 1：用 Dreamweaver CS 3.0 建立一个网站。

（1）启动 Dreamweaver CS 3.0，可以看到如图 1-4 所示界面，在该界面中用户可以快速读取最近使用过的文档，访问相关资源。



图 1-4　Dreamweaver CS 3.0 启动界面

（2）执行“站点”|“新建站点”菜单命令，则进入如图 1-5 所示的新建站点对话框。在图 1-5 中输入新建站点的名字如 mysite，和站点的 URL。单击“下一步”按钮。

（3）根据站点文件选择合适的环境，如果站点内仅包含 HTML 页面，可以选择“否，我不想使用服务器技术”，若包含 JSP 或 ASP 程序则可选择“是，我想使用服务器技术”，若选



图 1-5　新建站点对话框(编辑文件)

择后者，会显示如图 1-6 所示下拉框用于设定具体的服务器技术。此处选择“否，我不想使用服务器技术”，然后单击“下一步”按钮。



图 1-6　新建站点对话框(编辑文件，第 2 部分)

(4) 选择“编辑我的计算机上的本地副本，完成后再上传到服务器”设定所编辑的站点文件在本机上的存储位置，如图 1-7 所示，设置好后单击“下一步”按钮。

(5) 在弹出的对话框中设置连接到远程服务器的方式，此处选择“无”，如图 1-8 所示，然后单击“下一步”按钮。

(6) 在弹出的对话框中可以看到它显示出前面所设置的站点相关信息，如果没有问题则单击“完成”按钮，若需要修改则单击“上一步”按钮回到相应对话框进行修改，如图 1-9 所示。

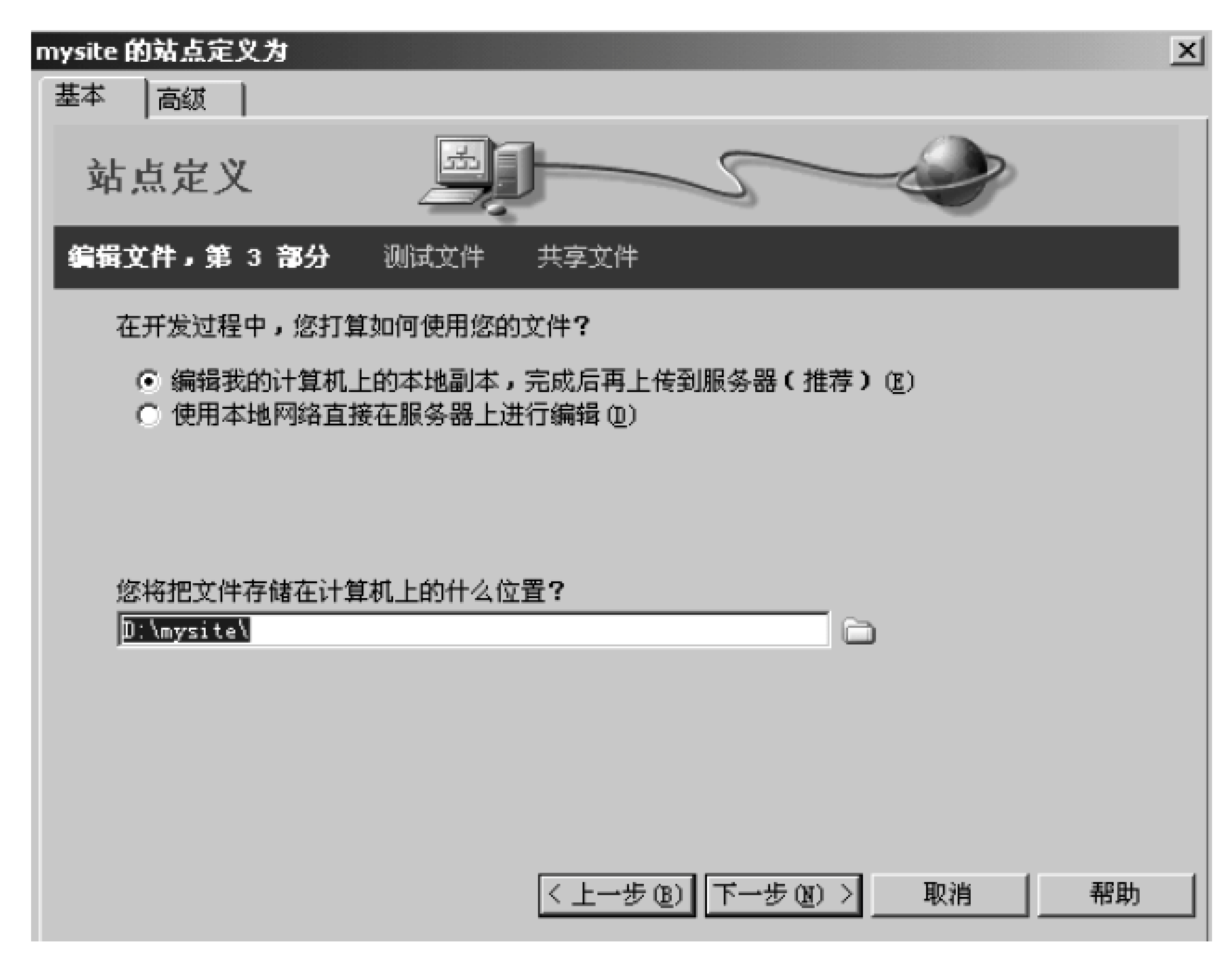


图 1-7 新建站点对话框(编辑文件,第 3 部分)



图 1-8 新建站点对话框(共享文件)



图 1-9 新建站点对话框(总结)

至此新建站点已经完成，可以在 Dreamweaver CS 3.0 的工作界面的面板组中的文件面板内看到新建的站点 mysite 已经显示在其中，如图 1-10 所示。若要管理该站点或其他已经建立的站点，可以选择菜单栏中的“站点”|“管理站点”选项，在弹出的对话框中可以看到已经创建的站点 mysite，如图 1-11 所示，可以对相关站点进行管理操作。



图 1-10　工作面板中显示新建站点 mysite



图 1-11　“管理站点”对话框

接下来可以在站点 mysite 中添加新的 HTML 页面。在文件面板右击站点 mysite，在弹出的快捷菜单中选择“新建文件”项，如图 1-12 所示。然后将生成的 HTML 页面文件名定义为所需的文件名，如 login.html，如图 1-13 所示。双击该文件，则在文档编辑区打开该文件的编辑窗口。



图 1-12　在 mysite 中新建文件(菜单)

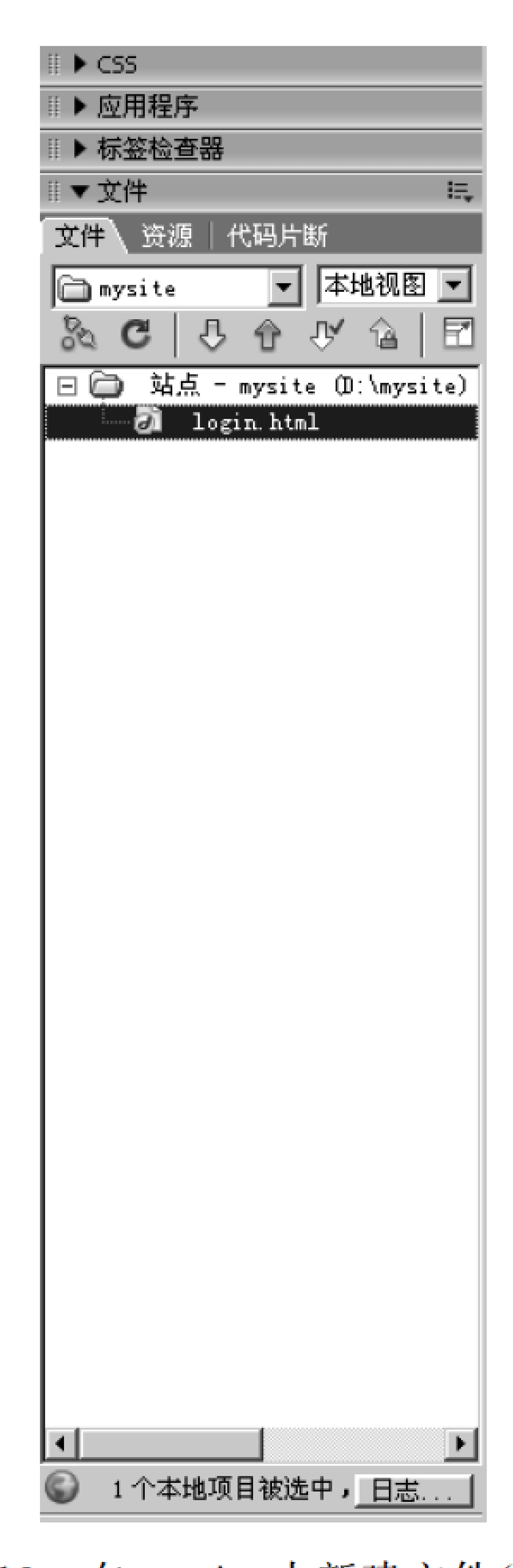


图 1-13　在 mysite 中新建文件(结果)

练习 2：利用 Dreamweaver CS 3.0 新建一个网页。

用 Dreamweaver CS 3.0 编写一个自选主题的网页，页面中布局结构和网页内容自行定义。要求：使用 HTML 页面排版、图片的嵌入、表格、表单等基本标记。

操作步骤略。

1.3 实验 1.2 使用 SharePoint Designer 2007 新建站点

实验目的：

(1) 熟悉 SharePoint Designer 2007 开发工具的功能。

(2) 了解和掌握 SharePoint Designer 2007 建设网站和开发网页。

实验内容：

(1) 熟悉 SharePoint Designer 2007 的开发环境，使用 SharePoint Designer 2007 新建一个站点。

(2) 用 SharePoint Designer 2007 开发一个简易的网页页面。

实验步骤：

练习 1：用 SharePoint Designer 2007 建立一个网站。

(1) 启动 SharePoint Designer 2007，然后执行“菜单”|“新建”|“网站”菜单命令，运行结果如图 1-14 所示。

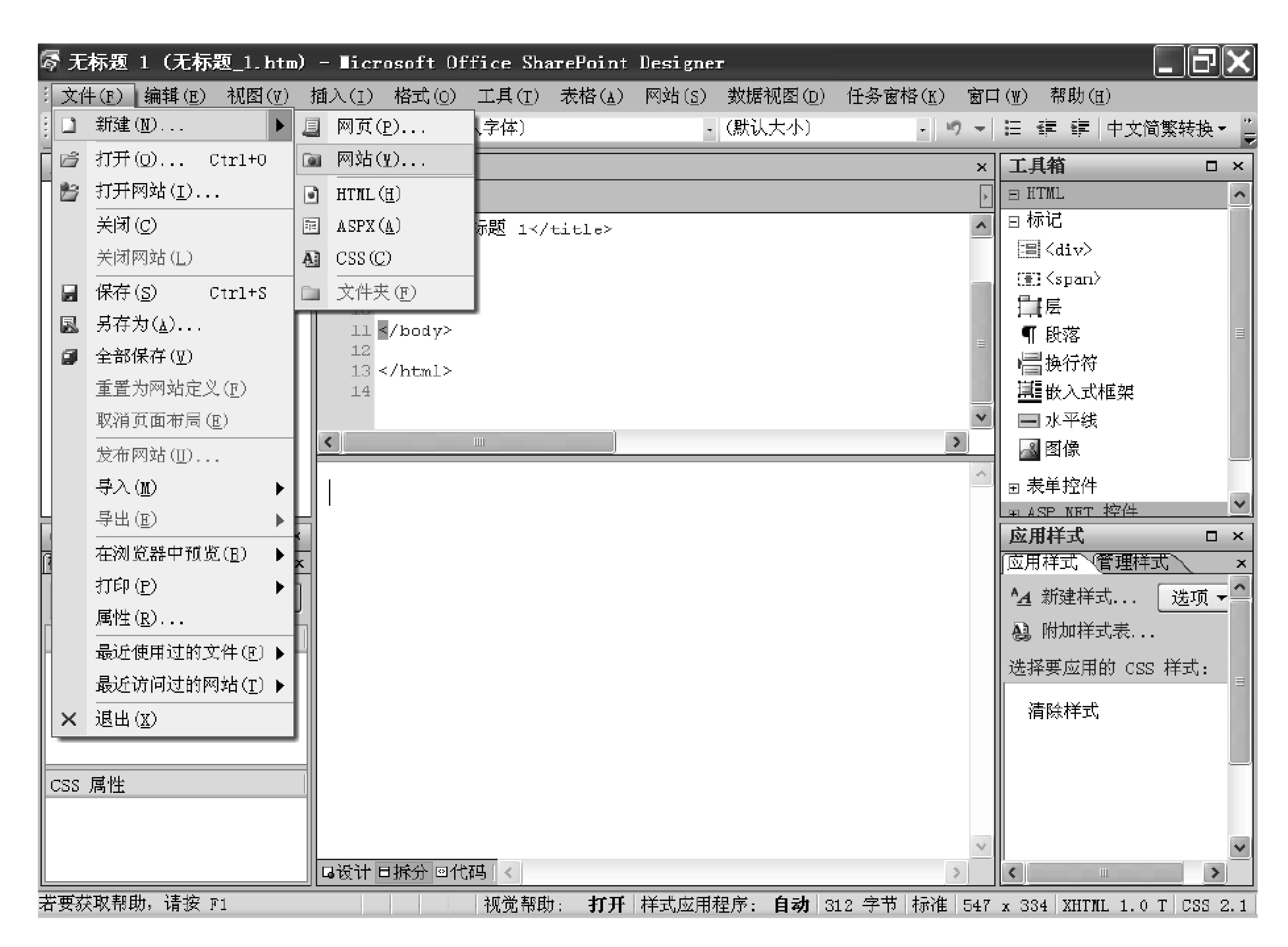


图 1-14 选择新建网站

（2）在弹出的“新建网站”界面中，选择“网站”项。在网站项下，有 3 种选择方式，用户可以根据要求自行选择。由于是了解新建网站功能，此处选择“常规”|“空白网站”方式建立网站，并设置保存网站文件的目录。运行结果如图 1-15 所示。

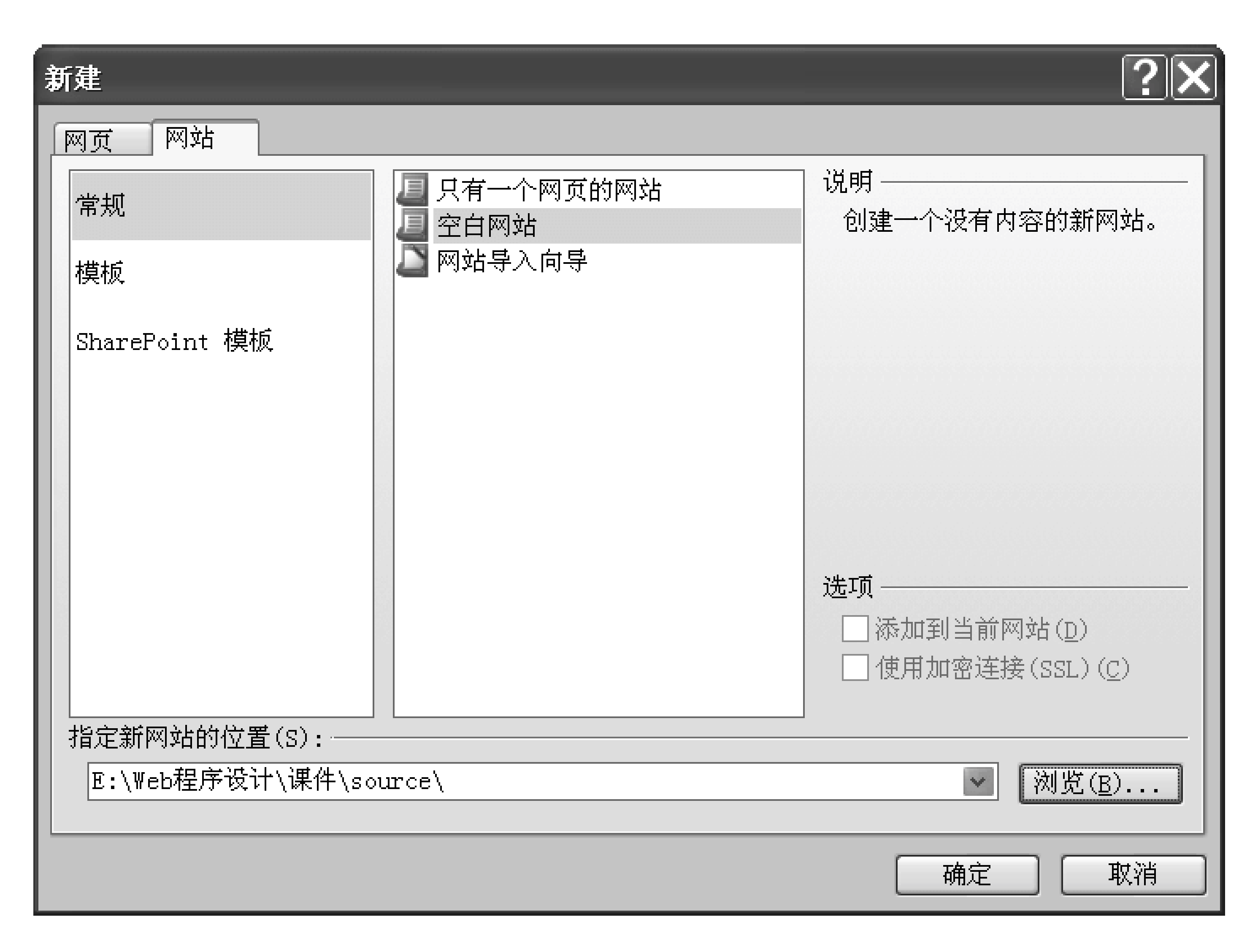


图 1-15　设置建设网站方式和路径

（3）网站建设完成后，会弹出“网站”窗口。可以通过右击鼠标，根据实际要求选择“新建”下的文件类型或子网站，建设网站的具体内容。运行结果如图 1-16 和图 1-17 所示。如果要对网站的文件进行建设，则可以在网站下双击具体文件就可以进入指定文件的编辑界面。

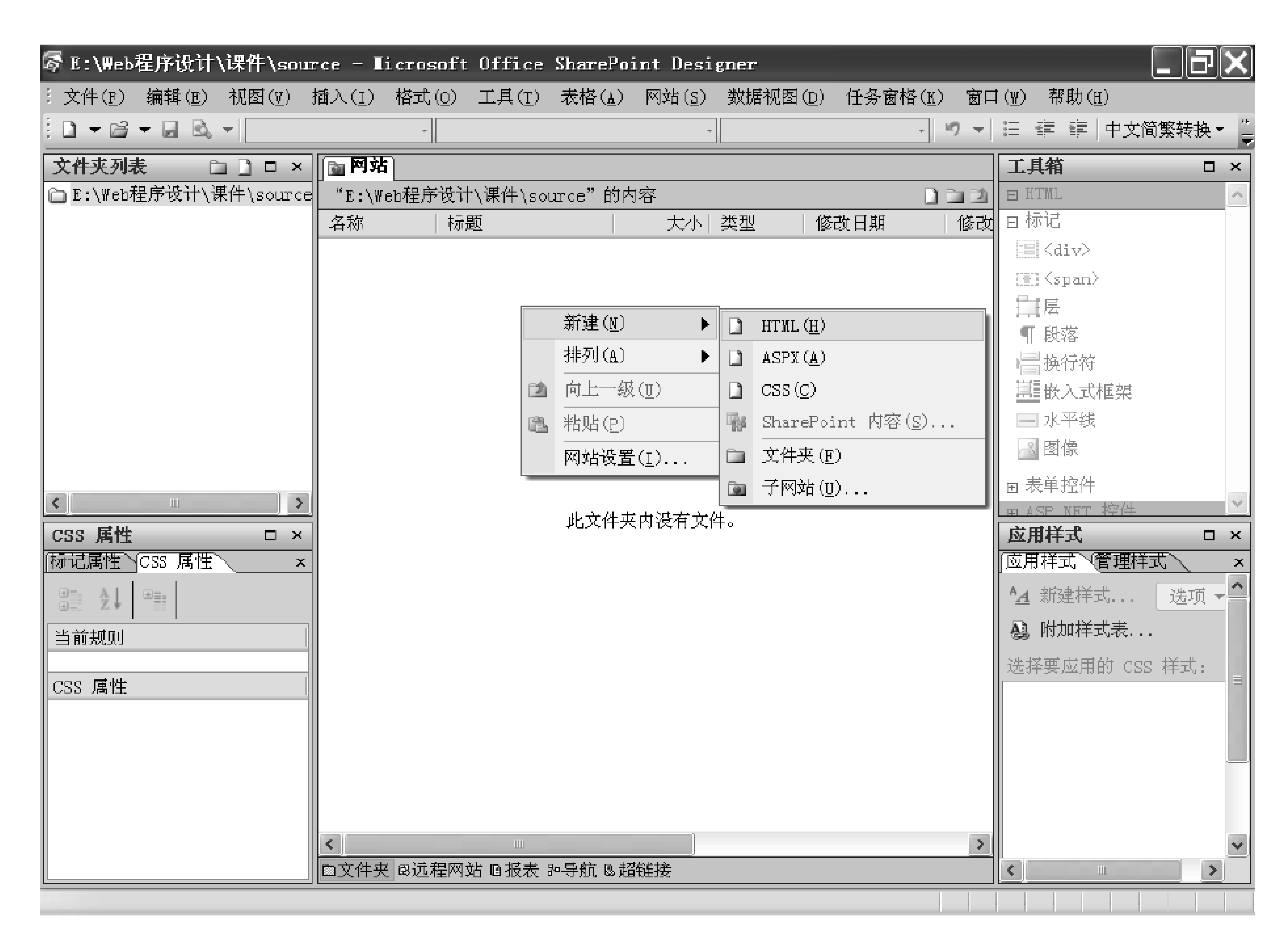


图 1-16　选择新建

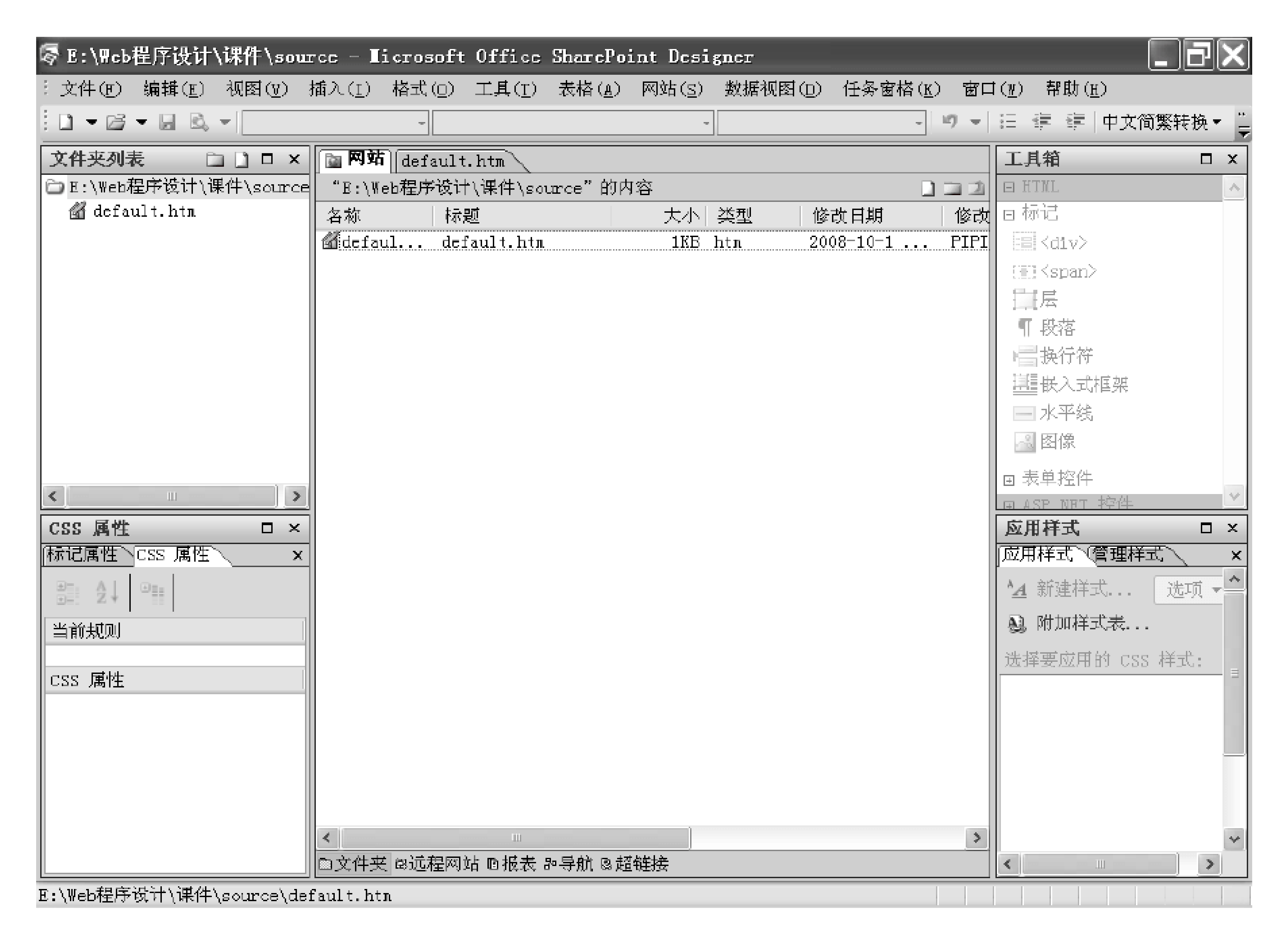


图 1-17　新建 HTML 文件成功

练习 2：用 SharePoint Designer 2007 开发一个网页页面。

请用 SharePoint Designer 2007 编写一个自选主题的网页，页面中布局结构和网页内容自行定义。要求：使用 HTML 页面排版、图片的嵌入、表格、表单等基本标记。比较 Dreamweaver CS 3.0 和 SharePoint Designer 2007 在建设网站和开发页面上功能的异同。

操作步骤略。

第 2 章　HTML 和 XHTML

步入 Internet 信息时代，网页已经成为人们进行信息交流的一种手段。而 HTML 与 XHTML 是编写网页的基础。通过对 HTML 和 XHTML 基本语法以及常见标签的了解，可以帮助用户开发形式生动、内容丰富的页面。在本章中，将针对 HTML 和 XHTML 的基本语法以及常见的标签设计相关的实验，引导读者进入设计和开发网页世界。

2.1　预备知识

2.1.1　HTML 的概述

HTML 表示超文本标记语言，是万维网的基本描述语言。HTML 可以支持文本、超链接、图像、多媒体、脚本语言和样式表等多种形式的内容。HTML 定义网页必须在网络浏览器中解释运行。目前 HTML 已经得到了广泛的支持和运用。

1．HTML 的页面结构

HTML 的页面用 html 标记定义，在页面内可以分成两部分：页面头、页面主体。其中页面头用 head 标记定义，而页面主体通过 body 标记定义。有时为了对页面的代码进行说明，往往用“＜！－－注释－－＞”定义注释。关于定义 HTML 页面结构的基本标记见表 2-1。

表 2-1　HTML 页面结构的基本标记

标　记	说　　明	标　记	说　　明
head	定义页面头	html	定义页面
title	定义页面的标题	body	定义页面主体
meta	定义元数据，具有属性： http-equiv：定义 HTTP 响应头 name：描述网页 content：说明页面内容	div	定义组合块
		span	定义文档的内联元素
		h1～h6	定义各种规格的头标题
hr	定义水平线	address	定义联络信息

2．HTML 的常见标记

除了与页面定义相关的标记，HTML 还根据实际需要定义了“文本处理标记”、“图像标记”、“链接标记”、“表格标记”、“表单标记”和“框架标记”等，这些标记的组合可以设计出生动的网页内容。这些标记的内容介绍见表 2-2。

3．HTML 的字符实体

为了表示特殊含义的内容，HTML 定义了字符实体。这些字符实体可以用两种形式表示“& 字符串”或“& 字符编号”。通过它们可以表达特定的字符信息。例如“ ；”和“&＃160；”表示空格。

表 2-2　HTML 的常见标记

类　型	标　记	说　　明	类　型	标　记	说　　明
文本处理	p	定义段落	列表处理	ul	定义无序列表
	pre	预定义文本		ol	定义有序列表
	em	定义等宽字体		li	定义列表项
	strong	定义强调字体		dl	定义定义列表
	dfn	定义字体		dt	定义描述项
	code	定义代码字体		dd	定义解释项
	samp	定义范例字体	表格处理	table	定义表格
	kbd	定义键盘字体		thead	定义表头
	var	定义变量字体		tbody	定义表的主体
	cite	定义引用字体		tfoot	定义表尾
	abbr	定义简写字体		caption	定义表的标题
	acronym	定义缩略字体		tr	定义表的行
	sup	定义上标字体		th	定义表头单元格
	sub	定义下标字体		td	定义表的单元格
	ins	定义被插入的文本	样式	style	定义内联样式
	del	定义已被删除的文本		font	定义字体
	blockquote	定义块引用		b	定义粗体文本
	br	换行		u	定义下划线文本
表单处理	form	定义表单		i	定义斜体文本
	input	定义输入域		tt	定义等宽文本
	textarea	定义文本域		big	定义大号字体
	select	定义选择列表		small	定义小号字体
	optgroup	定义选项组	框架处理	frameset	定义框架
	option	定义选项		frame	定义子窗口
	button	定义按钮		noframes	定义不支持框架内容
	fieldset	定义表单域		iframe	创建内联框架
	legend	定义表单域的标题	图像、对象和 Applet	img	定义图像
	label	定义标签		map	定义客户端图像映射
超链接	a	定义书签或链接		area	图像映射区域
	link	定义链接外部文件		object	定义嵌入对象
	base	定义当前文档的基址		applet	定义嵌入 applet
脚本	script	定义脚本			

2.1.2 XHTML 的概述

XHTML 表示扩展超文本标记语言，是在 HTML 4.01 的基础上，用 XML 1.0(扩展标记语言)改写而成的。这使得 XHTML 既具有 HTML 语言的特色，可以用同样的标记表示数据的内容，也具有 XML 语言的严格语法约束，方便定义数据。XHTML 能够结构化地表达数据，强化对模块的支持，符合数据内容与显示分离的发展趋势，是当前开发网页的主要标记语言。

XHTML 是在 HTML 的基础上发展而来，它具有 HTML 类似的标记定义、数据类型、字符实体、文件结构表示，但由于必须符合 XML 的严格语法要求，又体现出与 HTML 不同的特点。

(1) XHTML 文档必须通过 DOCTYPE 来声明文档类型定义。与 HTML 中文档类型定义的可有可无形成鲜明的对比。

(2) XHTML 文档的 html 标记中必须指定属性 xmlns，说明 XML 的命名空间是"http://www.w3.org/1999/xhtml"。如果在文档中忽略对 XML 命名空间的说明，系统会自动将上述的命名空间加入到文档中。

(3) 与 HTML 语法松散形成对比，XHTML 必须是良构的。这表示 XHTML 的标记可以嵌套，但是不能重叠。XHTML 的所有标记必须用小写字符表示。XHTML 的标记必须是封闭的，这表明即使是 XHTML 的空标记，也必须用空格与"/"作为结束，如"
"。此外，XHTML 标记的属性必须用显示的"="进行赋值，不能采用 HTML 中使用的简写形式赋值。

(4) XHTML 中取消了 HTML 中 name 和 id 属性定义标记名的混乱情况，而是统一用 id 属性区分不同的标记。

2.1.3 网站设计的基本要素

学习使用标记语言是设计和开发网站的基础。要建立一个高质量的网站还需要结合其他技术和知识，如美术、心理学等。通常开始建设一个网站，要充分考虑以下几个方面：

(1) 明确网站的客户定位。一个网站是否有用，并不是这个网站采用了多种技术手段，而是这个网站是否有人浏览访问，是否愿意接受服务。所以网站的客户定位就是明确网站提供服务或提供咨讯的群体对象。例如，老人、儿童、少年等不同的网站服务群体对象在使用网站的各个方面都有所不同。客户的定位在网站设计中首先要得到体现，因此，建立网站实质上是"建立以用户为中心"的网站。

(2) 确定网站的主题。确定网站的主题就是确定网站的内容。即网站的功能、作用、提供的服务或信息，这是网站开发的核心内容。如果网站的主题含糊不明确，客户会失去访问网站的信心。

(3) 确定网站的色彩体系。设计网站时要明确网站的色彩的选择和搭配。访问网站的第一观感并不是网站的内容和布局，而是色彩的搭配。合理的色彩选择和搭配会让网站突出主题，对网站浏览者的情绪会产生影响。色彩一般不宜太多，以两三种主题色彩为好。具体关于色彩体系(Color Schema)请参看有关资料，此处不再介绍。

(4) 遵循设计的基本原则。设计时应充分体现简洁直观。客户访问网站要非常容易达

到服务的内容，整个过程易学简单。另外，网站要快速，要充分考虑客户的计算机普遍较低配置，如果客户访问的页面，时间过长的话会影响访问网站的兴趣。这就使得设计人员在多媒体的应用中不能牺牲访问速度来实现复杂华而不实的效果，所以像 Flash 动画、视频等较大的文件要慎用。

(5) 建立良好的页面结构。页面结构明确网站网页具体的内容以及这些内容在网页的布局。良好的页面结构可以让用户快速找到需要的内容，而不是在各种链接中无所适从。要实现良好的页面结构，可以通过 logo、简洁高效的导航栏、高质量的站点链接等。

2.2 实验 2.1 HTML 基本标签的应用

实验目的：

(1) 了解 HTML 与 XHTML 文件的基本结构。

(2) 熟练掌握 HTML 和 XHTML 的常见标记。

(3) 掌握和比较 HTML 和 XHTML 语法要求的异同。

(4) 能根据实际要求开发和设计简单的静态网页。

实验内容：

(1) 分析用 HTML 编写的网页 Demo2_1. html，试运行该网页的运行结果。然后将该网页用 XHTML 语言改写。

(2) 阅读和运行网页 Demo2_2. html，指出并修改该网页中存在的错误。并说明原因。

(3)设计并开发一个“个人简历”的网页。

实验步骤：

选择开发网页的工具，按顺序依次完成上述的练习。

练习 1：了解 HTML 和 XHTML 网页的基本结构。

本练习帮助用户了解 HTML 文件与 XHTML 文件的基本结构。具体要求如下：

(1) 阅读和试运行 Demo2_1. html 网页，见程序清单 2-1。说出定义该网页所有标记的作用。

(2) 比较网页中使用的标记 span 和标记 div 的异同、比较标记 p 和标记 pre 的异同、比较标记 strong 和标记 big 的异同。

(3) 按照 XHTML 语法，将 Demo2_1. html 进行改写符合 XHTML 过渡类型的文件，并将改写好的文件保存为“Demo2_1. xhtml”，如果改写成功，则在网络浏览器中可以正常运行。

(4) 依次对 XHTML 改写的网页的文档类型分别设置为“严格类型”和“框架类型”，比较 3 种不同 XHTML 文档类型的不同之处。

程序清单 2-1：

```
<!--Demo2_1.html-->
<html>
```

```
<head>
  <title>第一个 HTML 网页</title>
  <meta name="keywords" content="网页基础,Web page">
  <meta name="author" content="陈轶,ChenYi,chenyi">
  <style type="text/css">
   <!--
     .textstyle{
               text-decoration:underline
               }
   -->
  </style>
</head>
<body align="center">
  <h3>了解<strong><big>H</big>TML</strong>和<strong><big>X</big>HTML</strong></h3>
  <p>
   <div class="textstyle">步入<b>Internet</b>信息时代,网页已经成为人们进行信息交流的
一种手段。</div>而<b>HTML</b>与<b>XHTML</b>是编写网页的基础。通过对<b>HTML</b>和<b
>XHTML</b>基本语法以及常见标签的了解,可以帮助用户开发生动内容丰富的页面。<br>在本章
中,将针对<b>HTML</b>和<b>XHTML</b>的基本语法以及常见的标签设计相关的实验,引导读者进
入设计和开发网页世界。
</p>
<pre>
   <span class="textstyle">步入<b>Internet</b>信息时代,网页已经成为人们进行信息交流的
一种手段。</span>
   而<b>HTML</b>与<b>XHTML</b>是编写网页的基础。通过对<b>HTML</b>和<b>XHTML </b>基
本语法以及常见标签的了解,可以帮助用户开发生动内容丰富的页面。在本章中,将针对<b>HTML</
b>和<b>XHTML</b>的基本语法以及常见的标签设计相关的实验,引导读者进入设计和开发网页
世界。
</pre>
<hr width="80%">
<address>
<p align="center">&quot;Web 程序设计实用教程 &quot;编写小组<br>
E-mail:<a href="mailto:somebody@ yahoo.com.cn">sombody@ yahoo.com.cn</a>
</p>
</body>
</html>
```

练习 2:比较 HTML 与 XHTML 文件的异同。

本练习的目的是比较 HTML 与 XHTML 文件的异同。网页 Demo2_2. html 是一个能显示“太阳系星系”的 XHTML 文件。显示的效果是,当用户用鼠标选中一个星球会显示该星球的名字。由于该文件在书写过程中按照 HTML 4. 01 的语法要求书写,导致网页无法在浏览器中正常显示。要求如下:

(1) 在 Dreamweaver CS 中打开 Demo2_2. html,将 Demo2_2. html 中修改文档类型说明为 HTML 的过渡类型,并保存为 Demo2_2. htm,并选择 Dreamweaver CS 的“窗口\结果”选项,打开“结果窗口”,选择“验证”项目,执行“验证当前文档”,如图 2-1 所示。观察验

证结果再运行，并记录运行结果。

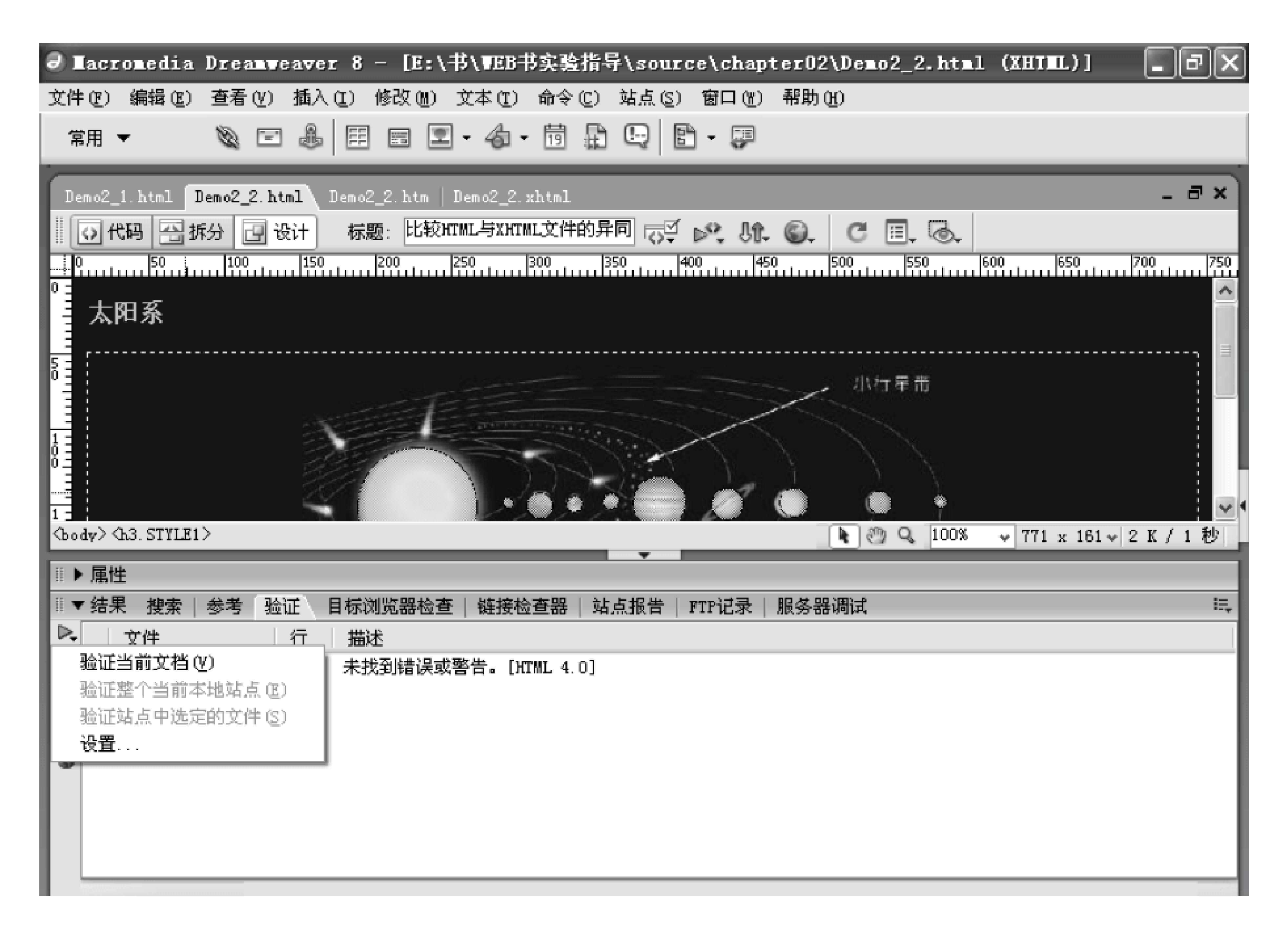


图 2-1　验证过程示意

（2）阅读 Demo2_2.html 文件，试找出该程序存在的问题。比较自行找出的问题与 Dreamweaver CS 的验证结果是否一致。请遵照 XHTML 语法要求，修改该文件使之能正常输出。要求运行结果与题意要求的显示效果保持一致。

程序清单 2-2：

```
<!--Demo2_2.html -->
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<title>比较 HTML 与 XHTML 文件的异同</title>
<style type="text/css">
<!--
body {
   background-color: #1D1D1F;
}
.STYLE1 {color: #CCCCCC}
-->
</style>
</head>

<body>
<h3 align="left" class="STYLE1">太阳系</h3>
```

```
<div align="center">
  <img src="image/sun.jpg" alt="太阳系示意图" useMAP="#sun">
   <MAP name="sun" id="sun">
     <Area shape="circle" coords="76,102,40" alt="太阳">
     <Area shape="circle" coords="136,99,4" alt="水星">
     <Area shape="circle" coords="157,101,9" alt="金星">
     <Area shape="circle" coords="180,100,6" alt="地球">
     <Area shape="circle" coords="204,98,6" alt="火星">
     <Area shape="circle" coords="236,99,18" alt="木星">
     <Area shape="circle" coords="281,100,11" alt="土星">
     <Area shape="circle" coords="325,99,10" alt="天王星">
     <Area shape="circle" coords="382,99,8" alt="海王星">
     <Area shape="circle" coords="422,98,5" alt="冥王星">
   </MAP>
</div>
</body>
</html>
```

练习 3：设计和开发"个人简历"网页。

通过本练习熟练运用 HTML/XHTML 标记开发和设计网页。本练习的内容是设计和开发一个"个人简历"的一个网页，要求：简历的内容包括"基本情况(包括姓名、出生年月、性别、电话、手机、个人照片等)、学习经历、工作经历、个人能力"；网页要明确网页的类型，对网页中部分标题选择特定头标题定义，不同部分有明显的段落或用换行分割。对于简历中有特色的部分用不同的字体突现强调。在网页最后部分要用 address 标记说明联系 E-mail 以及联系地址。开发网页的素材自行选定。

操作步骤略。

2.3 实验 2.2 列表和表格的设计

实验目的：

(1) 熟练掌握构成列表的标记以及相关属性的作用。

(2) 了解 3 种不同类型的列表，比较它们的不同用法。

(3) 了解构成表格的标记，以及应用表格的设计网页布局和数据显示。

实验内容：

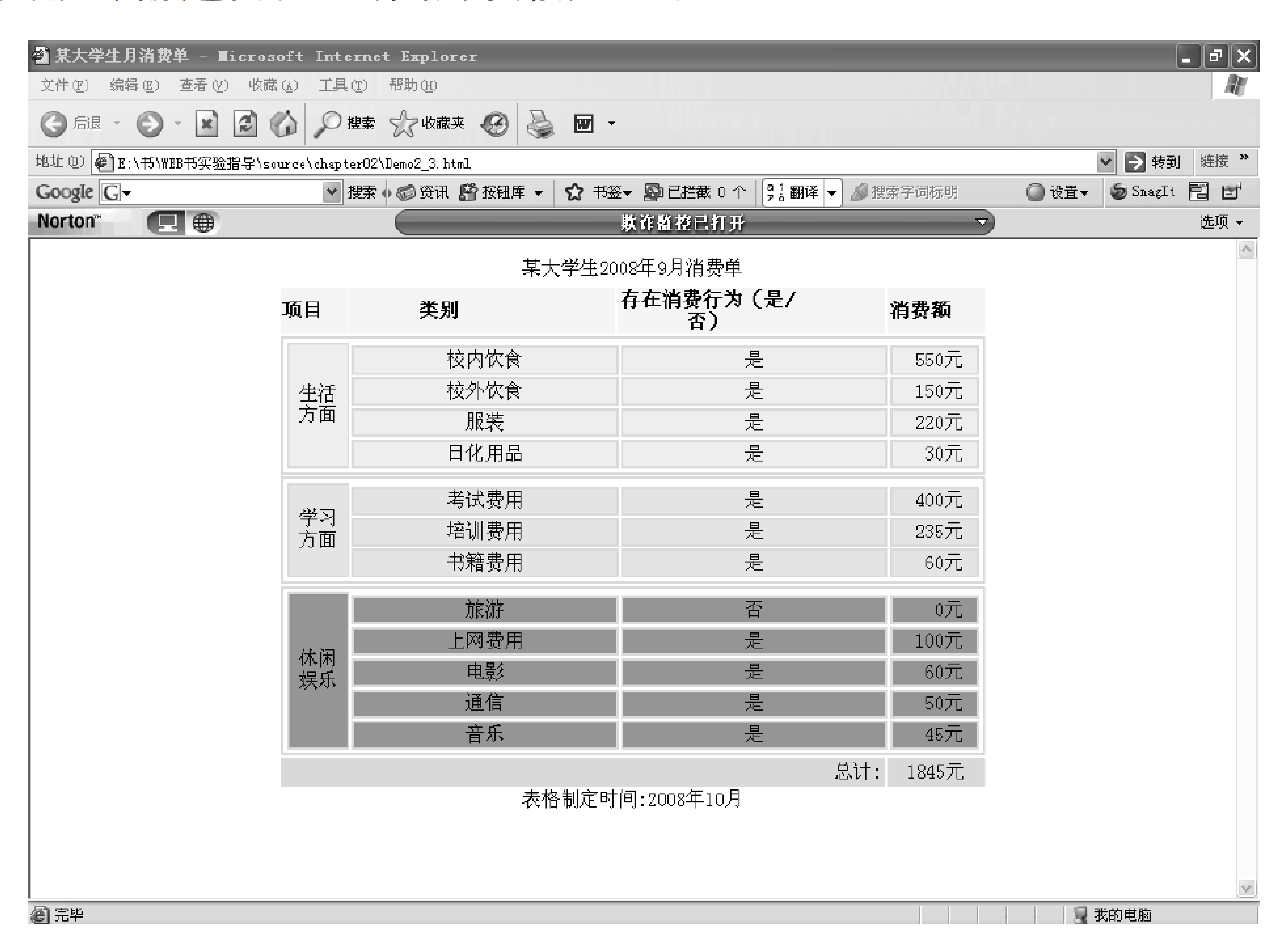
(1) 用表格定义数据。制作一个用户月消费表，具体包括的项目有生活方面(校内饮食、校外饮食、服装、日化用品、其他)，学习方面(考试费用、培训费用、书籍费用)，休闲娱乐方面(旅游、上网费用、电影、音乐、通信)这三方面。用户也可以参照自身情况编写。

(2) 设计一个网上书店主页面。页面的布局是传统的上下中左右型。要求用表格来设置布局。灵活运用列表来定义页面的内容。

实验步骤：

练习 1：用表格定义和表示数据。

本次练习的目的是了解制作表格的相关标记的使用。要求：提供网页的部分代码如程序清单 2-3（Demo2_3.html）所示，将该程序补充完整，使之能显示一个学生的月消费表（5 行 4 列表格）。该表格中定义 3 个项目：生活方面（校内饮食、校外饮食、服装、日化用品、其他），学习方面（考试费用、培训费用、书籍费用），休闲娱乐方面（旅游、上网费用、电影、音乐、通信）。此外，还需要显示制作表格的时间。具体数据可以参照自身情况定义。对于不同项目用不同颜色实现，运行结果类似图 2-2。



某大学生2008年9月消费单

项目	类别	存在消费行为（是/否）	消费额
生活方面	校内饮食	是	550元
	校外饮食	是	150元
	服装	是	220元
	日化用品	是	30元
学习方面	考试费用	是	400元
	培训费用	是	235元
	书籍费用	是	60元
休闲娱乐	旅游	否	0元
	上网费用	是	100元
	电影	是	60元
	通信	是	50元
	音乐	是	45元
		总计：	1845元

表格制定时间:2008年10月

图 2-2　月消费表示例图

程序清单 2-3：

```
<!--Demo2_3.html -->
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>某大学生月消费单</title>
</head>

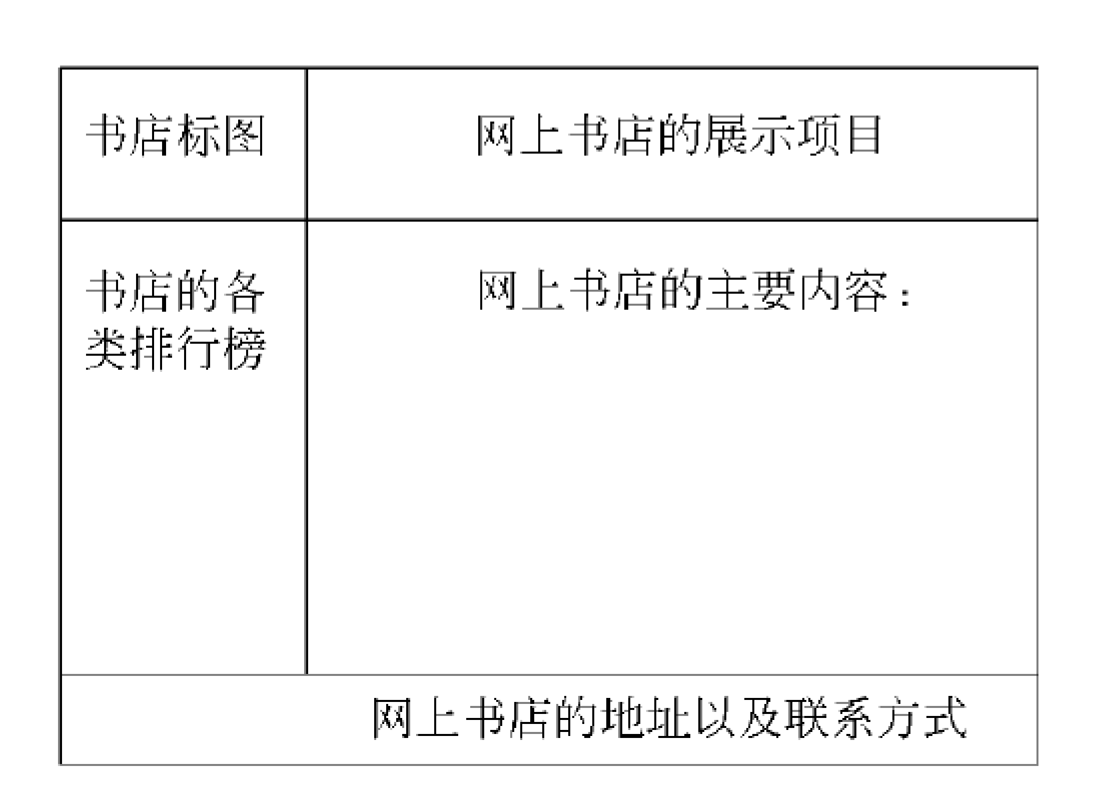
<body>
<p align="center">
  <table border="0" width="60%">
  <caption>
  某大学生 2008 年 9 月消费单
```

```
  </caption>
   <thead>
      <tr>
         <th width="35">项目</th>
         <th width="148">类别</th>
         <th width="149">存在消费行为(是/否)</th>
         <th width="53">消费额</th>
      </tr>
   </thead>
   <tfoot>
        <tr>
          <td colspan="3" bgcolor="#99FF99">
                <div align="right">总计:</div></td>
          <td bgcolor="#99FF99">1845元</td>
        </tr>
   </tfoot>
   <tbody>
        <!--补充代码,指定表格体-->
   </tbody>
  </table>
  <div align="center">表格制定时间:2008年10月</div>
</p>
</body>
</html>
```

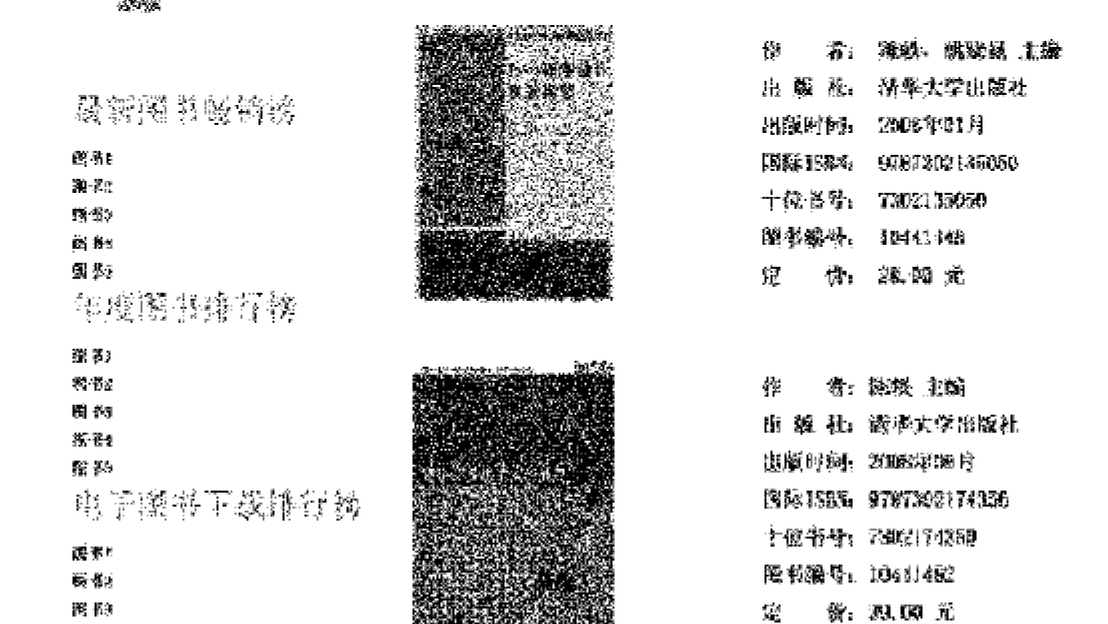
练习2:表格和列表实现网页布局。

本练习的主要目的进一步掌握表格的应用,并用表格定义网页的页面结构。此外,体会列表在网页中的应用。具体实验内容是,在Dreamweaver中新建一个网页文件Demo2_4.html,在这个网页中实现一个网上书店的展示书籍页面。要求:

(1) 网上书店的布局是上(左右)中(左右)下如图2-3所示,其中:上左部分插入书店的标图(即Logo);上右部分定义书店经营书商品的类别(文学、科技、少儿、教育、管理等);中



(a)



(b)

图2-3 网上书店的布局和运行示例

左部分定义书店的各类排行(如销售排行、人气排行、推荐排行等);在中右部分是网页主体,定义网上书店热销的各类书籍;下部定义书店的地址以及联系方式。具体内容和素材可以自行定义。

(2) 在网页的“上右”、“中左”、“中右”部分,分别用序列列表、无序列表和定义列表处理显示的内容。

思考:比较无序列表、有序列表以及定义列表 3 种列表的异同。请解释在网页中使用列表的原因?

操作步骤略。

2.4 实验 2.3 表单制作注册页面

实验目的:

(1) 了解制作表单常见组件(包括文本框、发送按钮、文本区等)的制作。

(2) 能熟练运用表单的相关标记解决实际问题。

实验内容:

要求为一个邮件服务网站提供一个注册新邮件的注册页面。对于该注册页面的内容需要指定:邮箱、密码、确认密码、昵称、性别、密码保护问题、密码保护问题的回答、密码保护问题的提示、选择兴趣组(旅游、文学、运动、音乐、电影、手机、财经、宠物等)和服务条款各项内容。

实验步骤:

在网页开发工具如 Dreamweaver CS 3.0 中打开已有的网页 Demo2_5.html,见程序清单 2-4,阅读实验内容,然后按照下列步骤完成实验内容指定注册表单的制作。

程序清单 2-4:

```
<!--Demo2_5.html-->
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>会员注册</title>
</head>

<body>
  <form name="frm1">
      <label>注册邮箱:</label>
      <label>密码:</label>
      <label>确认密码:</label>
      <label>密码保护问题</label>
      <label>回答</label>
      <label>密码保护问题的提示</label>
```

```
<label>昵称</label>
<label>加入兴趣组</label>
<label>服务条款：</label>
<label>我已经阅读并同意服务条款和隐私权政策。我对账户的使用遵守中华人民共和国法律。
</label>
<textarea name="service">
 本服务公约构成您与我们之全部协议,并规范您对于本服务之使用行为。本服务公约及您与
 我们之关系,均受到中华人民共和国法律管辖。我们未行使或执行本服务公约任何权利或规
 定,不构成对前述权利或权利之放弃。倘本服务公约之任何规定因与中华人民共和国法律抵
 触而无效,您依然同意应依照法律,努力使该规定所反映之当事人意向具备效力,且本服务公
 约其他规定仍应具有完整的效力及效果。本服务公约之标题仅供方便而设,不具任何法律或
 契约效果。
 </textarea>
  </div>
 </form>
</body>
</html>
```

(1) 为 Demo2_5.html 中,选择合理的组件将程序清单中没有定义的注册内容补充完整,并保存。

(2) 定义“发送”和“重置”按钮,让它们分别实现将注册的信息发送到指定页面和将注册信息全部清空的功能,观察运行结果。比较并回答按钮组件、发送按钮组件和重置按钮组件的不同。

(3) 如果希望表单能放置到一个表单域中,并定义一个表单标题“会员邮件注册”,运行结果类似图 2-4,请在 Demo2_5.html 中加入实现内容。

图 2-4　有表单域标题的注册页面

（4）将定义的会员注册表单在页面中设计合理的样式，使得表单显示合理，项目和组件能明显区分。

2.5　实验 2.4　多重框架和超链接

实验目的：

（1）了解和熟练运用定义多重框架的相关标记。

（2）比较多重框架与表格在设计网页布局的不同之处，了解二者的应用领域。

（3）了解和熟练掌握超链接的相关标记。

（4）运用超链接解决实际问题。

（5）初步了解设计和建立网站的关键要素以及过程。

实验内容：

（1）分别利用框架结构和表格结构为太阳系的八大行星及冥王星建立一个简易网站。

（2）自选一个主题（如个人网站、班级网站、学院网站、小型花店网站、奥运宣传网站等），根据选择的主题，设计和建设网站。

（3）访问 Nokia 中国网站（http://www.nokia.com.cn）、Nokia 香港网站（http://www.nokia.com.hk）、Nokia 全球网站（http://www.nokia.com），比较三个网站主页界面的异同。访问中国三星电子网站（http://www.samsung.com/cn）、香港三星电子网站（http://www.samsung.com.hk），比较它们网站主页界面的异同。并对 Nokia 中国网站和三星中国网站的首页的界面设计进行比较。

实验步骤：

本实验是由 3 个任务构成。

练习 1：建立介绍太阳系八大行星及冥王星的简易网站。

本次练习的目的是了解和运用框架结构和多链接的实现。已有网页 top.html、left.html 和main.html。其中，top.html 定义了关于该网站的 logo，代码见程序清单 2-5，left.html 定义了 10 个链接（主页、水星、金星、地球、火星、木星、土星、天王星、海王星、冥王星），分别链接 main.html 对应的书签，见程序清单 2-6，main.html 是对太阳系的一个简单介绍，并通过一个太阳系的图中设置行星的映射，实现对行星的快速导航。要求：

（1）请定义一个“上左右”型的框架网页 Demo2_6.html，如图 2-5。

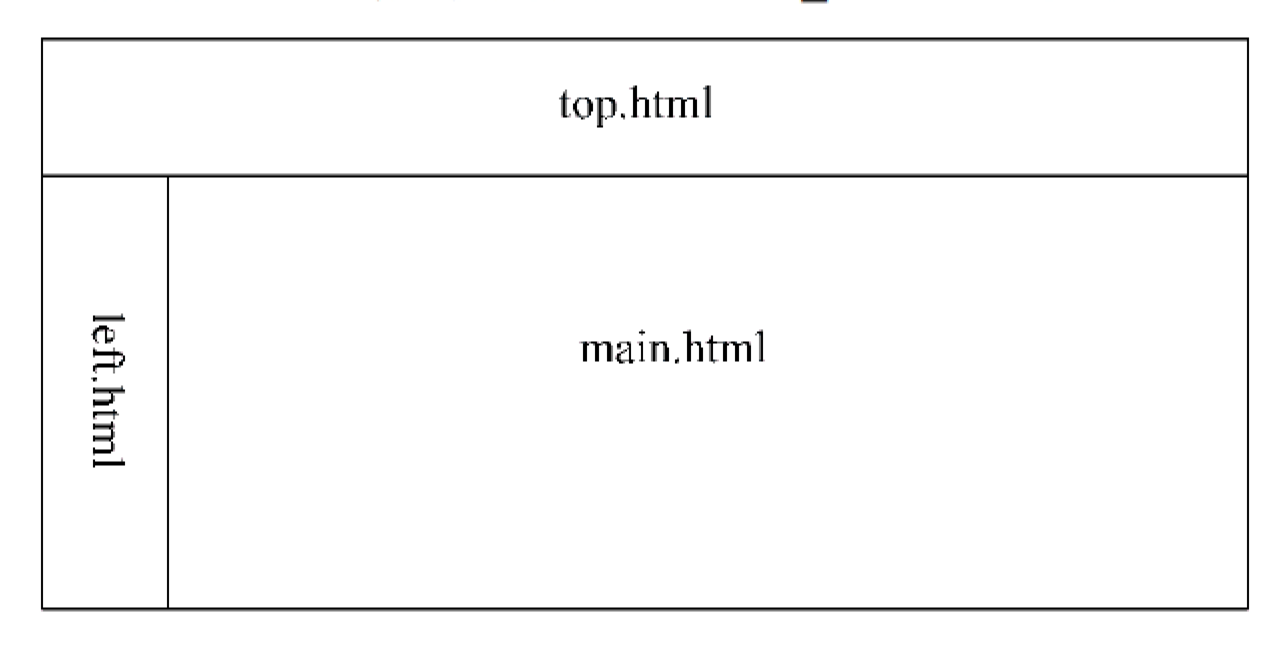


图 2-5　上左右框架示例

(2) 单击 left.html 的链接，观察运行结果。修改程序清单 2-6，使得单击链接后，能在新窗口打开目标内容。说明 left.html 中标记 a 的作用与 main.html 中标记 a 的作用。

程序清单 2-5：

```
<!--top.html-->
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>太阳系 Logo</title>
</head>

<body bgcolor="#1d1d1f">
<img src="image/solarlogo.jpg" alt="太阳系的 logo" width="216" height="79"/>
</body>
</html>
```

程序清单 2-6：

```
<!--left.html-->
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>菜单</title>
<style type="text/css">
<!--
body,td,th {
    color: #FF9900;
}
body {
    background-color: #1d1d1f;
}
a:link {
    color: #FF9900;
    text-decoration: none;
}
a:visited {
    color: #999999;
    text-decoration: none;
}
a:hover {
    text-decoration: none;
}
```

```
a:active {
     text-decoration: none;
}
-->
</style>
</head>

<body>
<p align="center">
<a href=" ">主页</a><br/>
<a href="main.html#start1">水星</a><br/>
<a href="main.html#start2">金星</a><br/>
<a href="main.html#start3">地球</a><br/>
<a href="main.html#start4">木星</a><br/>
<a href="main.html#start5">火星</a><br/>
<a href="main.html#start6">土星</a><br/>
<a href="main.html#start7">天王星</a><br/>
<a href="main.html#start8">海王星</a><br/>
<a href="main.html#start9">冥王星</a>
</p>
</body>
</html>
```

程序清单 2-7：

```
<!--main.html-->
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/
xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>主页</title>
<style type="text/css">
<!--
.STYLE1 {color: #FFFFCC}
p{color:#FF0000}
a:link {
     color: #FFCCCC;
     text-decoration: none;
}
a:visited {
     text-decoration: none;
     color: #FFCC00;
}
a:hover {
```

```
    text-decoration: none;
}
a:active {
    text-decoration: none;
}
body,td,th {
    color: #FF0000;
}
-->
</style>
</head>
<body bgcolor="#1d1d1f">
<div align="center">
<a name="main"/>
<img src="image/sun.jpg" alt="太阳系示意图" width="366" height="166" usemap="#sun"/>
<map name="sun" id="sun">
    <area shape="circle" coords="343,62,3" alt="冥王星" href="#start9"/>
    <area shape="circle" coords="307,63,6" alt="海王星"/>
    <area shape="circle" coords="263,64,11" alt="天王星" href="#start8"/>
    <area shape="circle" coords="228,63,7" alt="土星" href="#start7"/>
    <area shape="circle" coords="193,63,14" alt="木星" href="#start6"/>
    <area shape="circle" coords="166,63,4" alt="火星" href="#start5"/>
    <area shape="circle" coords="148,63,5" alt="地球" href="#start4"/>
    <area shape="circle" coords="128,64,6" alt="金星" href="#start3"/>
    <area shape="circle" coords="111,65,7" alt="水星" href="#start2"/>
    <area shape="circle" coords="62,66,32" alt="太阳" href="#start1"/>
</map>
</div>
<p><a name="start1">水星</a>
<br/>水星是太阳系中最靠近太阳的大行星。距太阳的平均距离约 58000000 千米。它是太阳系大行星中最小的,直径约 4870 千米,比冥王星大。  <a href="#main">返回</a>
</p>
<p>
<a name="start2">金星</a><br/>金星在距离太阳 1 亿零 7500000 千米的平均距离处,沿近圆轨道绕日公转。从随着金星一起绕日运行的地球上看,金星也呈现出和月球类似的相位变化。从大小和质量来看,金星和地球差不多。金星的直径只比地球小 653 千米,为 12103 千米;金星的质量约是地球的 81%。  <a href="#main">返回</a>
</p>
<p>
<a name="start3">地球</a><br/>按离太阳由近及远的次序为第三颗。它有一个天然卫星——月球。地球大约有 46 亿年的历史。不管是地球的整体,还是它的大气、海洋、地壳或内部,从形成以来就始终处于不断变化和运动之中。  <a href="#main">返回</a>
</p>
<p>
<a name="start4">火星</a><br/>火星是太阳系八大行星之一,按离太阳由近到远的顺序,火星排
```

```
在地球的后面，列为第四。它的平均直径为 6790 千米，约为地球直径的一半。它的密度也比地球小，为 3.933 克/立方厘米(地球为 5.52 克/立方厘米)。火星与太阳的平均距离为 228000000 千米，环绕太阳一圈约相当于地球上的 687 天。火星上的一天相当于地球上的 24 小时 37 分 22.6 秒，比地球的一天稍长一点儿。  <a href="#main">返回</a>
</p>
<p>
<a name="start5">木星</a><br/>木星是太阳系八大行星中最大的一颗，按离太阳由近到远的次序，它排第五。木星是夜空中最亮的几颗星之一，仅次于金星，通常比火星亮(除火星冲日时以外)，有时比最亮的恒星天狼星还亮。  <a href="#main">返回</a>
</p>
<p>
<a name="start6">土星</a><br/>土星是太阳系第六颗。土星在很多方面像木星，如它与木星同属于巨行星，它的体积是地球的 745 倍，质量是地球的 95.18 倍。在太阳系八大行星中，土星的大小和质量仅次于木星，占第二位。它像木星一样被色彩斑斓的云带所缭绕，并被较多的卫星所拱卫。它由于快速自转而呈扁球形。赤道半径约为 60000 千米。土星的平均密度只有 0.70 克/立方厘米，是八大行星中密度最小的。  <a href="#main">返回</a>
</p>
<p>
<a name="start7">天王星</a><br/>天王星的赤道半径约 25000 千米。体积约为地球的 65 倍，在八大行星中仅次于木星和土星。相当于地球质量的 14.63 倍。密度较小，每立方厘米只有1.24 克。因此，它虽然比海王星大，质量却只有海王星质量的 85%。在太阳系八大行星中，它的质量仅次于木星、土星和海王星，占第四位。  <a href="#main">返回</a>
</p>
<p>
<a name="start8">海王星</a><br/>海王星按离太阳远近排在第八，它的赤道直径为49528 千米，质量是地球的 17 倍，平均密度 1.64 克/立方厘米，体积是地球的 44 倍。它的轨道接近正圆，平均距离太阳 44.97 亿千米，是地球到太阳距离的 30 倍，绕太阳公转一周需要 164.79 年。  <a href="#main">返回</a>
<p>
<a name="start9">冥王星</a><br/>1930 年美国天文学家汤博发现冥王星，将它定义为大行星。冥王星与太阳的平均距离约为 59 亿千米。数十年来，科学家普遍认为太阳系有九大行星，但随着一颗比冥王星更大、更远的天体的发现，使得冥王星大行星地位的争论愈演愈烈。一是由于其发现的过程是基于一个错误的理论；二是由于当初将其质量估算错了，误将其纳入到了大行星的行列。因此在国际天文学联合会大会上，是否要给冥王星"正名"成为了大会的焦点，为此，天文学家给出了各种方案。
    2006 年 8 月 24 日，国际天文学联合会第 26 届大会，经两千余天文学家表决通过——太阳系只有八大行星。不再将传统九大行星之一的冥王星视为行星，而将其列入"矮行星"。
    因为根据"保守新行星定义"：一是必须围绕太阳运转的天体；二是质量足够大，能依靠自身引力使天体呈圆球状；三是其轨道附近应该没有其他物体。
    冥王星对最后一条条件不符，冥王星的轨道是和海王星有所交集的。  <a href="#main">返回</a>
</p>
</body>
</html>
```

练习 2：设计并建设一个自选主题的网站。

自选主题（如个人网站、班级网站、学院网站、小型花店网站、奥运宣传网站等），设计并建立一个网站。要求：按照 2.1.3 小节中介绍建设网站的基本要素来实现。网站的素材自行准备。要求书写一个设计报告，阐述建设网站的设计方案。

操作步骤略。

练习 3：了解商业网站的设计。

通过互联网络，访问 Nokia 中国网站（http://www.nokia.com.cn）、Nokia 香港网站（http://www.nokia.com.hk）、Nokia 全球网站（http://www.nokia.com），按照设计基本要素（客户对象、网站的主题、页面结构、色彩体系、文化背景、民族特色、设计风格等多方面）来比较三个网站首页界面设计的异同。并访问中国三星电子网站（http://www.samsung.com/cn）、香港三星电子网站（http://www.samsung.com.hk），比较二者的异同。并对 Nokia 中国网站和三星中国网站进行比较。根据网上调查的结果书写报告，说明网站开发的基本要素的作用，并以 Nokia 和三星网站的实例进行说明。

操作步骤略。

第 3 章　CSS 技术

CSS(Cascading Style Sheets,层叠样式表)是对 Web 页面显示效果进行控制的一套标准。CSS 扩充了 HTML 标记的属性设定,使得页面显示效果更加丰富,表现效果更加灵活,更具有动态性。同时使用 CSS 可以将页面样式定义和 HTML 文件分离,使得页面开发及维护工作更易进行。

3.1　预备知识

3.1.1　CSS 基本语法

(1) CSS 定义的基本语法格式如下:

```
选择符{规则列表}
```

其中选择符是指要使用该样式的对象,它可以是一个或多个 HTML 标记、CLASS 选择符或 ID 选择符,如果为多个则使用逗号“,”进行分隔。规则列表是由一个或多个属性定义语句组成的样式规则,各语句间使用分号“;”进行分隔。属性定义语句的语法格式为:

“属性名:属性值”

(2) 在页面文件中对 CSS 的定义有以下几种方式:

① 直接在页面文件中使用 HTML 标记的 style 属性。其语法格式如下:

```
<标记名 style="样式属性名 1:属性值 1;
              样式属性名 2:属性值 2;
              …">
```

② 在页面文件中定义内部样式表。其语法格式如下:

```
<style type="text/css">
<!--
    选择符 1,选择符 2,…{样式属性名 1:属性值 1;
                      样式属性名 2:属性值 2;
                      …}
         ⋮
    -->
</style>
```

③ 在页面文件中嵌入外部样式表。其语法格式如下:

```
<style type="text/css">
<!--
  @import url("外部 CSS 样式表文件名");
```

```
-->
</style>
```

④ 链接外部样式表。其语法格式如下：

```
<link type="text/css"
rel="stylesheet"
href="外部 CSS 样式表文件名">
```

(3) CSS 的注释语句。CSS 的注释语句是位于“/ * ”和“ * /”标记之间的语句内容。

3.1.2 CSS 选择符

CSS 选择符主要有 HTML 标记、CLASS 选择符和 ID 选择符 3 种。它们的定义和使用方法见表 3-1。

表 3-1 CSS 选择符的定义和使用

选 择 符	语 法 格 式	样式使用范围说明
HTML 标记	定义语法：标记{…} 使用语法：＜标记＞	在 HTML 文件中，所有该标记包含的文本都具有定义的 CSS 样式
CLASS 选择符	定义语法：* . 类名{…}或 . 类名{…} 使用语法：＜标记 class＝类名＞	在 HTML 文件中的所有使用该类名的标记都具有定义的 CSS 样式
	定义语法：标记. 类名{…} 使用语法：＜标记 class＝类名＞	在 HTML 文件中的所有指定该类名的该标记都具有定义的 CSS 样式
ID 选择符	定义语法：# ID 名{…} 使用语法：＜标记 id＝ID 名＞	在 HTML 文件中的所有使用该 ID 名的标记都具有定义的 CSS 样式
	定义语法：标记 # ID 名{…} 使用语法：＜标记 id＝ID 名＞	在 HTML 文件中的所有指定该 ID 名的该标记都具有定义的 CSS 样式

注意，这 3 种 CSS 选择符可以混合使用。除了这 3 种基本选择符外，CSS 中还提供了伪类。伪类是一类特殊的选择符，它和类选择符不同，不能由用户自己命名，而是由 CSS 定义，具有特定含义。伪类的基本语法如下：

```
选择符名：伪类名{属性名：属性值}
```

如定义 a：visited{color：# 3300FF；text-decoration：none；}，则表示在页面中使用超链接标记＜a＞时，当超链接被访问过后则将使用该样式。

也可以将类选择符和伪类混用，其基本语法如下：

```
选择符名.类名：伪类名{属性名：属性值}
```

如：

```
a.red:visited{color:# FF0000;text-decoration:none;}
```

要使用该样式可用下面语句

```
<a class=red href="# ">…</a>
```

常用的伪类见表 3-2。

表 3-2　CSS 常见的伪类

伪　类	说　　明	例
锚 a 元素的伪类	link 表示动态链接的未访问的链接状态 visited 表示动态链接的已访问链接状态 hover 表示动态链接的鼠标放在链接上的状态 active 表示动态链接的激活链接的状态	a:linked{color:green;} a:visited{color:red;} a:hover{color:blue;} a:active{color:black;}

first-letter	CSS 2.0 定义的首字母伪类,定义首字母的样式	p:first-letter{
		font-size:30px;
		}

first-line	CSS 2.0 定义的首行伪类,定义首行的样式	div:first-line{font-size:12px;}

注意:

(1) 和 a 标记相关的伪类分别表示超链接在 4 种不同的状态下的显示效果:link(未访问过的超链接)、visited(已访问过的超链接)、active(单击时的超链接)和 hover(鼠标停留在超链接上)。对于这 4 个伪类在定义时要注意它们的顺序,要按照 a:link,a:visited,a:hover,a:active 的顺序定义。

(2) 和文字段落相关的两个伪类分别为 first-letter 和 first-line。first-letter 定义了第一个字符的状态,而 first-line 定义了段落第一行的状态。

3.1.3　样式表的层叠顺序

可以使用 CSS 语句对 HTML 标记设置不同的显示样式,但是由于 HTML 标记在使用中常常有嵌套情况出现,那么对于控制同一页面内容的嵌套标记,究竟哪一个样式起作用,可以按照以下规则进行判断。

(1) 直接在页面文件中使用 HTML 标记的 style 属性定义的内联样式优先级最高。

(2) 其他的样式定义按照在页面文件中出现的顺序,越后出现的优先级越高。

(3) 由于 id 选择符一般最后定义,所以 id 选择符的优先级高于 class 选择符。

(4) 没有被定义样式控制的内容将使用浏览器的默认样式。

3.1.4　CSS 基本属性

CSS 的基本属性主要包括背景属性、文本属性、字体属性、边界属性、边框属性、边距属性、列表属性和定位属性等。

(1) CSS 背景属性主要包括 background-color、background-image、background-repeat、background-attachment 和 background-position 等。

(2) CSS 文本属性主要包括 text-indent、text-align、vertical-align、line-height 和 letter-spacing 等。

(3) CSS 字体属性主要包括 font-family、font-style、font-size、font-weight、font-variant、text-decoration 和 text-transform 等。

(4) CSS 边界属性主要使用 margin 属性来控制元素边界与网页其他内容的水平和垂

直间距，除此以外也可以使用 margin-top、margin-right、margin-bottom 和 margin-left 给 4 个边界单独设置具体属性值。

（5）CSS 边框属性主要有 border、border-style 和 border-color。

（6）CSS 边距属性主要使用 padding 来设置元素的内容与元素边框之间的距离，也可以使用 padding-top、padding-right、padding-bottom 和 padding-left 分别设置上、右、下、左 4 个方向的属性值。

（7）CSS 列表属性主要使用 list-style 来设置文字列表属性，控制列表的符号和位置，也可以通过 list-style-type、list-style-position 和 list-style-image 单独进行设定。

（8）CSS 定位属性主要有 top、left、position 和 z-index。

3.2　实验 3.1　CSS 选择符的使用

实验目的：

（1）掌握 CSS 的基本语法。

（2）掌握 CSS 选择符的使用方法。

实验内容：

编写程序使用 CSS 的 3 种选择符（HTML 标记、CLASS 选择符和 ID 选择符）对 HTML 标记进行样式定义，并观察程序运行结果。

实验步骤：

新建一个 HTML 文件 Css_1.html，在其中输入程序清单 3-1 的代码并保存。为使程序运行结果符合实验任务规定的要求，将程序中的 代码 1 ～ 代码 3 补充完整，并在浏览器中查看程序运行结果是否和图 3-1 一致。

此处定义font标记的样式：蓝色隶书，字体大小为20pt

此处使用CLASS选择符myfont1定义a标记的样式，点击链接到搜狐网主页

此处使用ID选择符myfont2定义p标记的样式

图 3-1　实验 3.1 运行结果

程序清单 3-1：

```
<!--Css_1.html 源代码-->
<html>
<head>
  <title>实验 1 CSS 选字符的使用</title>
</head>

<style type="text/css">
```

```
<!--
    .myfont1{
                font-style:italic;
                color:red;
                text-decoration:none
                }
    #myfont2{
                font-family:黑体;
                font-size:16px;
                letter-spacing:3px
                }
-->
</style>

<body>
<center>
  <font 代码 1 ><!--定义蓝色隶书字体为 20pt-->
     此处定义 font 标记的样式：蓝色隶书，字体大小为 20pt
  </font>
  <br>
<br>
  <a href="www.sohu.com"  代码 2 ><!--定义使用 myfont1-->
     此处使用 CLASS 选择符 myfont1 定义 a 标记的样式，单击链接到搜狐网主页
  </a>
  <br>
  <p  代码 3 ><!--定义使用 myfont2-->
     此处使用 ID 选择符 myfont2 定义 p 标记的样式
  </p>
</center>
</body>
</html>
```

3.3　实验 3.2　制作菜单

实验目的：

（1）进一步熟悉 CSS 的基本语法。
（2）掌握 CSS 的样式定义方法。

实验内容：

编写 CSS 样式表制作出如图所示的菜单效果。当打开浏览器后可见如图 3-2 所示的菜单样式，当鼠标移至不同菜单项上时，该菜单项的显示样式将发生变化，如当鼠标移至“实验内容”上时，出现如图 3-3 所示效果。

首页	课程简介	实验内容	教学资源	在线交流	联系我们

图 3-2　实验 3.2 运行结果(一)

首页	课程简介	实验内容	教学资源	在线交流	联系我们

图 3-3　实验 3.2 运行结果(二)

实验步骤：

新建一个 HTML 文件 Css_2.html，在其中输入程序清单 3-2 的代码，并将 代码段 1 和 代码段 2 处样式定义补充完整，使得该 HTML 文件在浏览器中显示如图 3-2 所示效果。

程序清单 3-2：

```
<!--Css_2.html 源代码-->
<html>
<head>
<title>菜单</title>
<style type="text/css">
<!--
 * {  margin:0;
      padding:0;
      border:0;
   }

body{
    font-family: arial, 宋体, serif;
    font-size:12px;
}

#menu{
    line-height: 26px;
    list-style-type: none;
}

#menu a{
      display: block;
      width: 80px;
      text-align: center;
}

#menu a:link{
```

```
        color:#000000;
        text-decoration: none;
}

#menu a:visited{
        color:#3300FF;
        text-decoration:none;
}

#menu a:hover{
        color: #FFFFFF;
        text-decoration: none;
        font-weight: bold;
}

#menu li{
    代码段 1 <!--设置菜单项-->
}

#menu li a:hover{
    代码段 2 <!--设置鼠标放在链接上的背景-->
}
-->
</style>
</head>

<body>
<ul id="menu">
<li>
    <a href="#">首页</a>
</li>
<li>
    <a href="#">课程简介</a>
</li>
<li>
    <a href="#">实验内容</a>
</li>
<li>
    <a href="#">教学资源</a>
</li>
<li>
    <a href="#">在线交流</a>
```

```
</li>
<li>
  <a href="#">联系我们</a>
</li>
</ul>
</body>
</html>
```

说明：在这个实验中，注意到在定义 CSS 样式时使用到了上下文选择符(contextual selector)，如本程序中的 #menu a{display: block; width: 80px; text-align:center;}。所谓上下文选择符是由两个或更多的选择符组成，这些选择符之间以空格隔开，它的含义是只有当最后一个选择符(如上例中的“a”)是第一个选择符(如上例中的 ID 选择符 menu)的直接后代时该样式才起作用。

3.4 实验 3.3 使用 CSS 样式设置页面布局

实验目的：

(1) 熟悉运用 CSS 统一站点风格的技巧。

(2) 了解如何使用 CSS 美化页面布局。

实验内容：

设计 CSS 样式表将已有的 HTML 页面(未使用 CSS 样式的 HTML 文件，运行效果如图 3-4 所示)按照图 3-5 的效果布局并显示。

《Web程序设计》课程教学网

- 首页
- 课程简介
- 实验内容
- 教学资源
- 在线交流
- 联系我们

课程简介

本课程的先修课程是计算机基础、高级语言、计算机网络。通过对本课程的教学，使学生了解Web应用程序的开发技术，包括Internet基础知识、脚本语言，静态页面和动态页面的设计以及服务器端程序的编制和数据库的应用；掌握交互式Web系统的设计方法；能读懂较复杂的交互式系统原代码，并且能应用所学知识开发建设小型的网站。

>>详细内容

实验内容

在学习《Web程序设计》课程的基础上，通过实验来提高学生对所学知识的了解和程序开发能力。本课程实验内容从应用和实践角度出发，结合相关教学内容，帮助学生熟悉HTML基本标记、脚本语言、CSS基本技术、JSP基本语法和数据库编程等知识点。实验内容可基本分为客户端程序开发和服务器端程序开发两部分。

>>详细内容

E-mail：******@***.com

*****大学计算机系《Web程序设计》教学小组@2008

图 3-4 未使用样式表文件的 Css_3.html 运行结果



图 3-5　使用样式表文件 Css_3.css 后的 Css_3.html 应达到的效果

实验步骤：

（1）新建 Css_3.html 文件，在其中输入程序清单 3-3 所示的代码。

程序清单 3-3：

```
<!--Css_3.html 源代码-->
<html>
  <head>
    <title>CSS 实验 3</title>
</head>

  <body>
    <div>
      <h1>《Web 程序设计》课程教学网</h1>
      <ul>
        <li>
          <a href="#">首页</a>
        </li>
        <li>
          <a href="#">课程简介</a>
        </li>
        <li>
          <a href="#">实验内容</a>
        </li>
        <li>
          <a href="#">教学资源</a>
        </li>
        <li>
```

```
        <a href="#">在线交流</a>
      </li>
      <li>
        <a href="#">联系我们</a>
      </li>
   </ul>
  </div>

  <div>
  <div>
    <h3>课程简介</h3>
  </div>

  <div>
      <p>
        本课程的先修课程是计算机基础、高级语言、计算机网络。通过对本课程的教学,使学
      生了解 Web 应用程序的开发技术,包括 Internet 基础知识、脚本语言,静态页面和动态页面
      的设计以及服务器端程序的编制和数据库的应用;掌握交互式 Web 系统的设计方法;能读懂
      较复杂的交互式系统源代码,并且能应用所学知识开发建设小型的网站。
      </p>
    <p><a href="#">>>详细内容</a></p>
  </div>
  </div>

  <div>
  <div>
    <h3>实验内容</h3>
  </div>
  <div>
    <p>
      在学习《Web 程序设计》课程的基础上,通过实验来提高学生对所学知识的了解和程序开
    发能力。本课程实验内容从应用和实践角度出发,结合相关教学内容,帮助学生熟悉 HTML 基
    本标记、脚本语言、CSS 基本技术、JSP 基本语法和数据库编程等知识点。实验内容可基本分为
    客户端程序开发和服务器端程序开发两部分。
    </p>
      <p><a href="#">>>详细内容</a></p>
  </div>
  </div>

  <div>
    <p>E-mail:******@ ***.com</p>
    <p>******大学计算机系《Web 程序设计》教学小组@ 2008</p>
  </div>
 </body>
</html>
```

（2）在同一目录下新建 Css_3.css 文件，该文件的功能是对 Css_3.html 文件进行样式定义。将 Css_3.css 文件链接到 Css_3.html 中，使得 Css_3.html 中的 HTML 标记使用所定义的样式达到图 3-5 所示效果。请大家按要求完成 Css_3.css 文件代码的编写，并在程序清单 3-3 的代码中添加样式使其在浏览器中运行得到图 3-5 效果。

说明：本实验中对 HTML 页面的设计中使用了很多 div 标记，它将页面的内容分割成不同部分，这样可以方便地对每一部分内容进行样式定义。可以使用＜div id＝"IdSelector_name"＞或＜div class＝"ClassSelector_name"＞来将已经定义好的样式用于 div 标记所包含的区域中。

第4章　客户端脚本语言

应用于客户端 Web 程序设计的脚本语言弥补了 HTML 语言的不足,大大增强了客户端 Web 页面的动态性和交互性。JavaScript 是目前较为常用的脚本语言,它既可以用于客户端 Web 程序开发,也可以用于服务器端 Web 程序开发,是一种嵌入 HTML 的脚本语言,它不需要编译,在客户端可以通过浏览器解释执行。

4.1　预备知识

4.1.1　JavaScript 基本语法

(1) 将 JavaScript 代码嵌入 HTML 文件的方法有两种。

① 直接在 HTML 文件中使用 HTML 中的<script>…</script>标记。其描述形式如下:

```
<script language="JavaScript">
JavaScript 语句段
</script>
```

或

```
<script type="text/javaScript">
JavaScript 语句段
</script>
```

② 将 JavaScript 程序代码和 HTML 文件分别编写,并将 JavaScript 程序代码以扩展名.js 保存,然后在 HTML 文件中通过<script>标记将指定的 JavaScript 文件导入进来。其描述形式如下:

```
<script src="JavaScript 文件路径">
```

(2) JavaScript 区分大小写。

(3) JavaScript 中使用换行符作为一条语句的结束标志。如果需要将几条语句放在同一行书写,可以在各条语句间使用分号(;)进行分隔。

(4) JavaScript 中注释语句的描述形式有两种,可以使用“//”或“/ * … * /”标记 JavaScript 的注释语句。

4.1.2　JavaScript 常见的数据类型

JavaScript 提供了 4 种基本的数据类型,分别是数值型、字符串型、布尔型和空值。

(1) 数值型数据包括整数和浮点数。

(2) 字符型数据是用双引号(" ")括起来的字符串或单引号(' ')括起来的字符。

(3) 布尔型数据可以取 true 和 false 两个值。

(4) 空值 null。

4.1.3 变量和常量

1. 变量

(1) 命名规则。JavaScript 中变量名必须是以字母或下划线(_)开头，由字母、数字和下划线共同组成的字符串。

(2) 声明方法。在 JavaScript 中变量的声明方法有两种。

① 使用关键字 var 声明变量。描述形式如下：

```
var 变量名
```

或

```
var 变量名=变量值
```

② 不使用关键字 var，直接使用赋值语句声明变量。描述形式如下：

```
变量名=变量值
```

(3) 作用域。JavaScript 中变量也可以分为全局变量和局部变量两类。全局变量定义在所有函数体外，对任何函数可见；局部变量是定义在某一函数体内，只对该函数可见。

2. 常量

JavaScript 中常量可以是整型、布尔型、字符型等，如 123、21.34、"this is an apple"等。

4.1.4 运算符

JavaScript 中运算符可以分为赋值运算符、算数运算符、逻辑运算符、关系运算符、字符串运算符和位运算符 6 类，具体见表 4-1。

表 4-1 JavaScript 运算符表

<table>
<tr><th>分　类</th><th>运　算　符</th></tr>
<tr><td>赋值运算符</td><td>＝、＋＝、－＝、＊＝、/＝和%＝</td></tr>
<tr><td>算数运算符</td><td>＋、－(二元运算)、＊、/、%、＋＋、－－和－(一元运算)</td></tr>
<tr><td>逻辑运算符</td><td>&&、||和！</td></tr>
<tr><td>关系运算符</td><td>＝＝、!＝、>、>＝、< 和<＝</td></tr>
<tr><td>字符串运算符</td><td>＋</td></tr>
<tr><td>位运算符</td><td>&、|、～、^、<<和>></td></tr>
</table>

4.1.5 对象和数组

1. 对象

JavaScript 中对象包含属性和方法两部分。属性就是对象的物理信息的描述，而方法指该对象可以进行的操作。要使用对象，必须确保该对象已经存在。对对象中属性的访问可以通过“对象名.属性名”的方式进行，同样，要调用对象中定义好的方法可以通过“对象名.方法名”的形式实现。

在 JavaScript 中要自定义对象，可以使用关键字 function 来声明。其语法形式如下：

```
function 对象名(属性列表){
this.属性 1=参数 1
this.属性 2=参数 2
        ⋮
this.方法 1=函数 1
this.方法 2=函数 2
        ⋮
}
```

2. 数组对象

在 JavaScript 中,可以通过使用其内置对象 Array 来创建数组对象。创建具体数组对象实例的方法如下:

```
var 数组名=new Array()
```

或

```
var 数组名=new Array(数组长度)
```

Array 对象常用的属性有 length,用来描述数组的长度;常用的方法有 join()、reverse()和 sort()方法。

4.1.6 函数

1. 函数定义

在 JavaScript 中函数的定义要使用关键字 function,其基本的语法形式如下:

```
function 函数名(参数 1, 参数 2, …, 参数 n){
    函数体语句段
    return 表达式
}
```

2. 函数调用

定义了函数后就可以调用该函数执行已经定义好的操作。函数的调用方法很简单,只需要给出已定义的函数名和要传递到函数中的参数值就可以。

4.1.7 JavaScript 的控制流程

1. 条件语句 if…else

条件语句的基本结构有 if、if…else、嵌套 if…else 多分支语句和 switch…case 语句。

2. 循环语句

JavaScript 中的循环语句有 for 语句、while 语句、do…while 语句、for…in 语句和 with 语句等。其中 for…in 语句用来对已知对象的所有属性进行循环操作的循环语句,它不需要使用计数器来控制循环次数,也无须知道对象的属性个数就可以进行相关操作。

4.1.8 JavaScript 的事件处理

JavaScript 具有事件驱动的特点。JavaScript 中的事件可以理解为用户和 Web 页面

的交互操作，通常是由鼠标或热键的动作引发。对事件进行处理的函数称为事件处理程序(event handler)或事件处理方法。由于客户端 JavaScript 通常是嵌套在 HTML 文件中，所以其事件处理一般会将 HTML 页面的具体元素和定义好的事件处理方法关联起来。

4.1.9 JavaScript 的内置对象

JavaScript 常用的内置对象有 Array、String、Math 和 Date 对象等。其中 String 对象是针对字符串的操作定义的对象，Math 对象是针对常数和数学运算定义的对象，Date 对象是针对日期和时间操作定义的对象。

4.2 实验 4.1 简易计算器

实验目的：

(1) 熟悉 JavaScript 的基本语法。

(2) 掌握 JavaScript 函数的定义和使用。

实验内容：

使用 JavaScript 编写程序实现计算器功能，可以实现浮点数的加、减、乘、除运算。

实验步骤：

新建一个 HTML 文件 JS_1.html，在其中输入程序清单 4-1 的代码并保存。运行该程序将在浏览器中显示如图 4-1 所示页面，在其中单击按钮输入"3.5+2.8"，则显示如图 4-2 所示页面，再单击"="，则将显示如图 4-3 所示结果。单击按钮"C"，则清除输入文本框中之前输入的所有数据。为使程序运行结果符合实验任务规定的要求，将程序中的 代码1 ～ 代码3 补充完整。

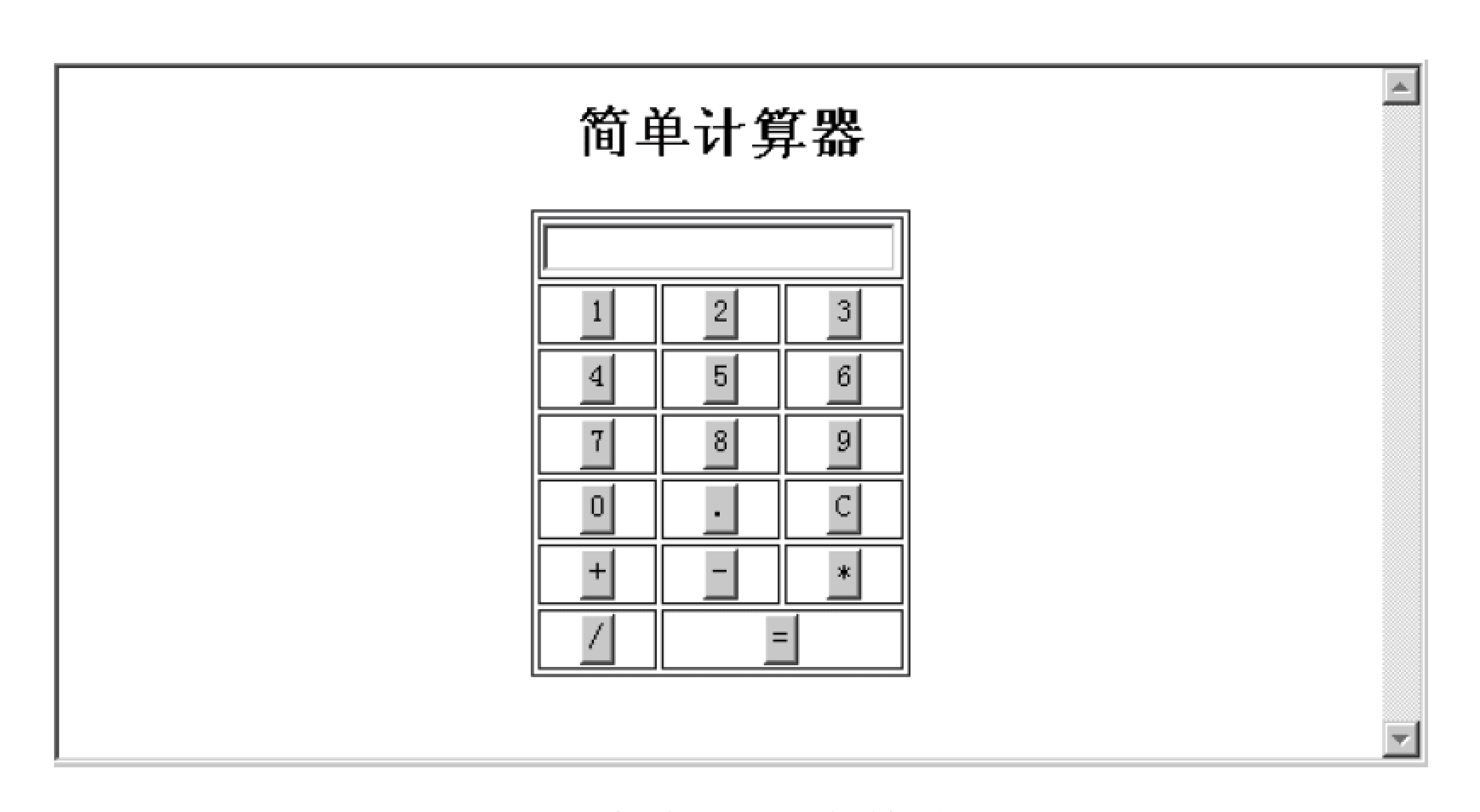


图 4-1 实验 4.1 运行效果(一)

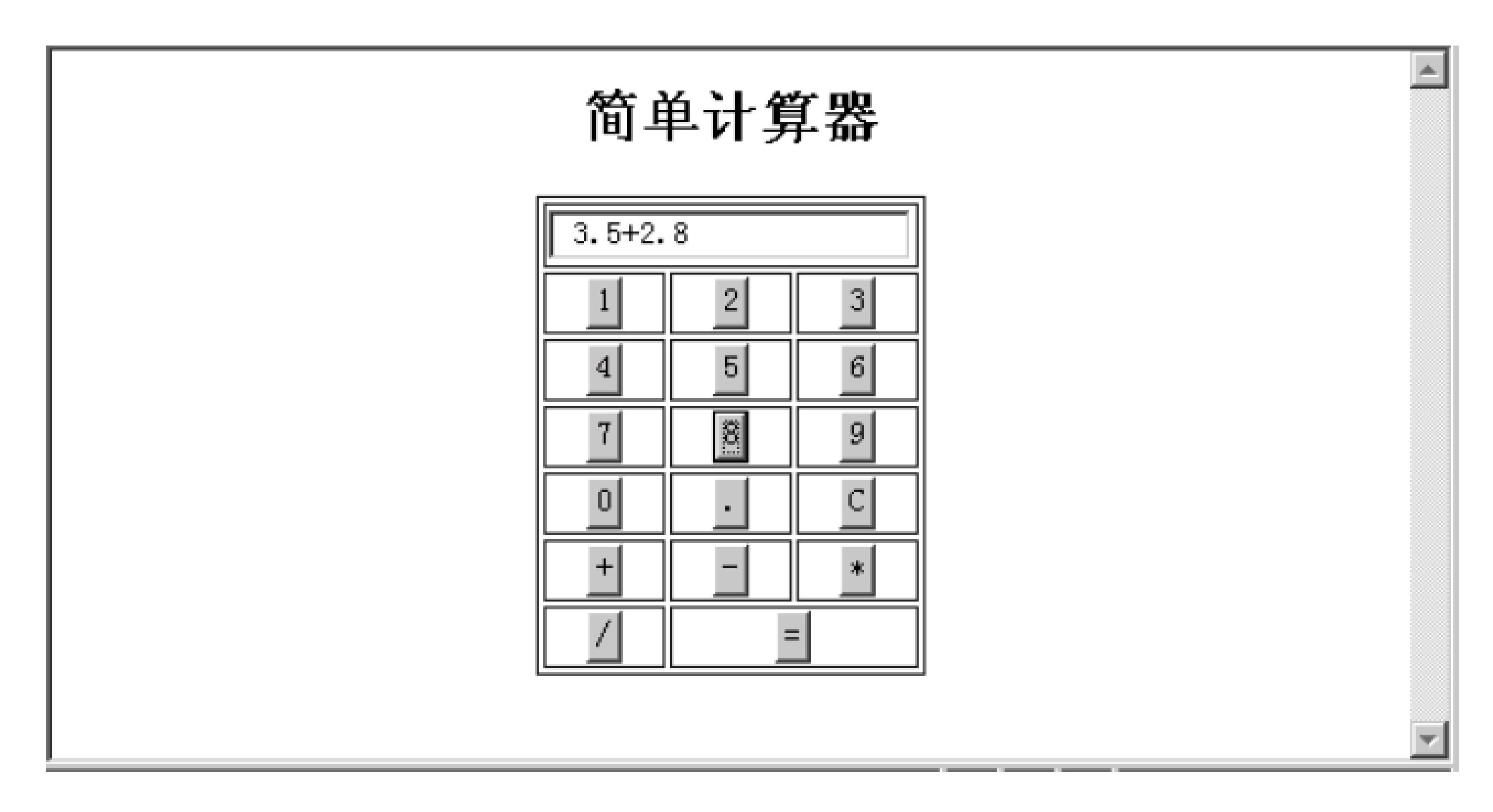


图 4-2　实验 4.1 运行效果(二)

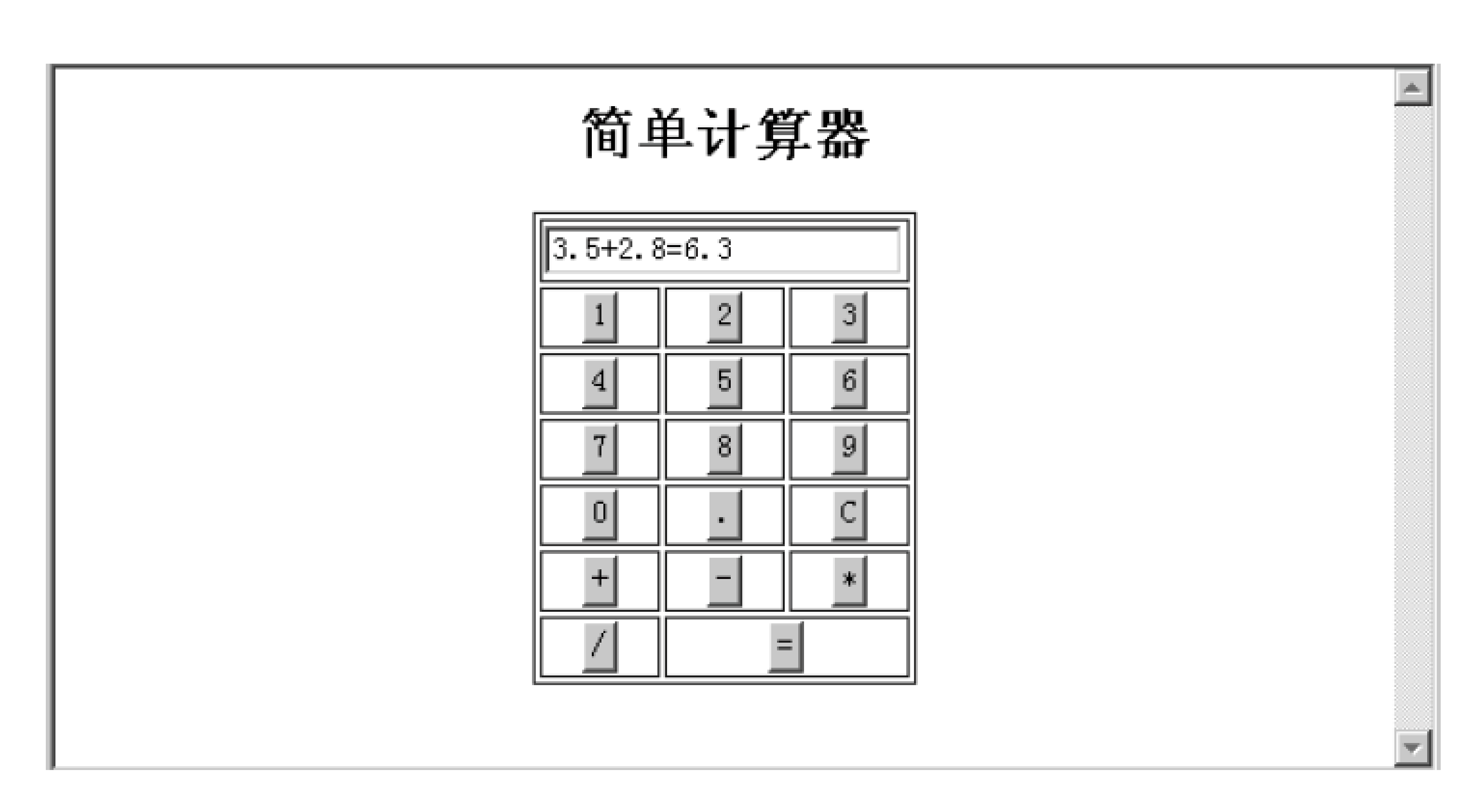


图 4-3　实验 4.2 运行效果(三)

程序清单 4-1：

<!--JS_1.html 源代码-->

```
<html>
<head><title>简单计算器</title>
<script language="JavaScript">
<!--
function SetExp(str){
//设置文本框内显示的字符串
    代码 1
}
function Cal(){
//计算输入的算术表达式并在文本框内给出结果
    代码 2
}
function Clear(){
//清空文本框内的输入字符串
    代码 3
}
```

```
-->
</script>
</head>
<body>
<center>
<h2>简单计算器</h2>
<form name=Nform method=get>
<table border="1" bordercolor="#003300">
<tr>
    <td colspan="3">
        <input type=text name="result">
    </td>
</tr>
<tr align="center">
    <td>
        <input type="button" value="1" onClick="SetExp('1')">
    </td>
    <td>
        <input type="button" value="2" onClick="SetExp('2')">
    </td>
    <td>
        <input type="button" value="3" onClick="SetExp('3')">
    </td>
</tr>
<tr align="center">
    <td>
        <input type="button" value="4" onClick="SetExp('4')">
    </td>
    <td>
        <input type="button" value="5" onClick="SetExp('5')">
    </td>
    <td>
        <input type="button" value="6" onClick="SetExp('6')">
    </td>
</tr>
<tr align="center">
    <td>
        <input type="button" value="7" onClick="SetExp('7')">
    </td>
    <td>
        <input type="button" value="8" onClick="SetExp('8')">
    </td>
    <td>
        <input type="button" value="9" onClick="SetExp('9')">
    </td>
```

```
</tr>
    <tr align="center">
    <td>
        <input type="button" value="0" onClick="SetExp('0')">
    </td>
    <td>
        <input type="button" value="." onClick="SetExp('.')">
    </td>
    <td>
        <input type="button" value="C" onClick="Clear()">
    </td>
</tr>
<tr align="center">
    <td>
        <input type="button" value="+" onClick="SetExp('+')">
    </td>
    <td>
        <input type="button" value="-" onClick="SetExp('-')">
    </td>
    <td>
        <input type="button" value="*" onClick="SetExp('*')">
    </td>
</tr>
<tr align="center">
    <td>
        <input type="button" value="/" onClick="SetExp('/')">
    </td>
    <td colspan="2">
        <input type="button" value="=" onClick="Cal()">
    </td>
    </tr>
</table>
</form>
</center>
</body>
</html>
```

思考：本实验中按钮“C”的功能是清除文本框内输入的所有字符，如果需再添加一个按钮“Backspace”，可以每单击该按钮一次只清除在文本框内最近输入的一个符号，如在文本框内已输入“3.52”，单击 Backspace，则文本框内显示字符串变成“3.5”，该按钮功能应该如何实现？

4.3 实验 4.2 鼠标跟踪

实验目的：

（1）进一步掌握 JavaScript 的基本语法和 JavaScript 函数的定义和使用。

（2）熟悉 JavaScript 的事件处理。

实验内容：

使用 JavaScript 编写程序实现一行文字跟随鼠标移动的效果。

实验步骤：

新建一个 HTML 文件 JS_2. html，在其中输入程序清单 4-2 的代码并保存。为使程序运行结果符合实验任务规定的要求，将程序中的 代码 1 ～ 代码 5 补充完整。

程序清单 4-2：

```
<!--JS_2.html 源代码-->

<html>
<head>
<title>鼠标跟踪</title>
<STYLE type=text/css>
.mystyle {COLOR:red;
        FONT-FAMILY: 宋体;
        FONT-SIZE: 20pt;
        POSITION:absolute;
        TOP:100px;
        WIDTH:200px;
        Z-INDEX:50;
        VISIBILITY: visible
}
</STYLE>
<script language="JavaScript">
<!--
    var x,y;
    var flag=0;

    function MouseMove(){
      x=document.body.scrollLeft+event.clientX;
    //获取当前鼠标位置的 X 坐标
      y=document.body.scrollTop+event.clientY;
    //获取当前鼠标位置的 Y 坐标
      flag=1;
    }

    function SetPos(){
    if( 代码 1 ){
    //当鼠标移动时，重新设置图层位置
```

```
        div1.style.posLeft=x+20;
        //设置图层位置的 X 坐标
        div1.style.posTop=y;
        //设置图层位置的 Y 坐标
        }
        代码 2
        //每 100ms 刷新一次图层位置坐标
        }
    -->
    </script>
    </head>
    <body onLoad=" 代码 3 " onmousemove=" 代码 4 ">
    <div id="div1" class=" 代码 5 ">
    鼠标跟踪演示
    </div>
    </body>
    </html>
```

思考：本实验中实现了文字跟随鼠标移动的效果，如果需要实现图片跟随鼠标移动应该将程序如何进行修改？

4.4 实验 4.3 JavaScript 控制 CSS

实验目的：

（1）进一步掌握和运用 JavaScript 开发动态网页。

（2）了解 JavaScript 控制 CSS 样式表。

实验内容：

本次已知有 3 个 CSS 样式表文件，分别规定显示网页段落的文字的尺寸的样式为 small、media 和 large。要求定义一个 JavaScript 文件控制这些样式文件，使得网页为用户提供变换网页内文字的大小的功能。

实验步骤：

已知网页 demo.html（代码见程序清单 4-3），是一个显示介绍 JavaScript 页面，另定义“大”、“中”和“小”按钮，用户可以单击按钮达到修改文字大小的功能。其中 3 个 CSS 样式表文件的 small.css、media.css 和 large.css，分别定义文字的样式为 small、media 和 large。它们的源代码分别见程序清单 4-4、程序清单 4-5、程序清单 4-6。网页 3 个按钮的控制是调用 change.js 的相关函数实现的。请将程序清单 4-7 中的 change.js 程序的 代码 1 ～ 代码 3 补充完整，实现修改网页文字大小的功能。具体的运行结果见图 4-4 和图 4-5。



图 4-4　修改样式后的显示大号字效果



图 4-5　修改样式后的显示小号字效果

程序清单 4-3：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<!--demo.html-->

<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
<link href="medium.css" rel="stylesheet" type="text/css" id="cssfile"/>
<!--导入默认的 medium.css-->
```

```
<script language="JavaScript" src="change.js" type="text/javascript"/>
<!--导入 change.js-->
<title>Javascript 控制 CSS 样式</title>
</head>

<body>
<div>
<h1>Javascript</h1><br/>
<div id="button">字体
    <input id="button0" type="button" value="小"/>
    <!--定义文字大小控制按钮 -->
    <input id="button1" type="button" value="中"/>
    <!--定义文字大小控制按钮 -->
    <input id="button2" type="button" value="大"/>
    <!--定义文字大小控制按钮 -->
</div>
</div>

<div id="paragraph">
<p>
JavaScript 是目前较为常用的脚本语言,它既可以用于客户端 Web 程序开发,也可以用于服务器端
Web 程序开发,是一种嵌入 HTML 的脚本语言,它不需要编译,在客户端可以通过浏览器解释执行。
</p>
</div>
</body>
</html>
```

程序清单 4-4:

```
/* small.css */
p {
    font-size: small;
}
```

程序清单 4-5:

```
/* medium.css */
p{
    font-size:medium;
}
```

程序清单 4-6:

```
/* large.css */
p{

    font-size:large;
}
```

程序清单 4-7：

```
//change.js
var button={};
//定义 button 对象；
button.addEvent=function(){
//定义 button 对象的添加事件函数
var buttons=document.getElementById("button").getElementsByTagName("input");
//获取所有的按钮
for (i=0; 代码 1 ;i++){
//为所有的按钮定义单击事件的处理函数
     buttons[i].onclick=function(){
             button.setSize(this.id.substring(6));
             //调用设置文字大小
             };
    }
}
button.setSize= 代码 2 {
//修改文本的样式
    if(style==0)
      document.getElementById("cssfile").href="small.css";
    else if(style==1)
      document.getElementById("cssfile").href="medium.css";
    else
      document.getElementById("cssfile").href="large.css";
}
window.onload=function(){
    代码 3
//加载并调用 button 对象的 addEvent 函数
    }
```

思考：如果有一个已知 HTML 文件以及定义不同的 CSS 样式文件，请考虑如何实现用户可以根据实际要求加载不同的 CSS 样式。

第 5 章　可扩展标记语言 XML

XML 作为一种半结构化的语言，继承了结构语言具有良好数据表示能力的特点。由于单纯的 XML 语言并没有实际意义，XML 往往与其他技术相结合，开发具有现实意义的应用。本章将侧重于 XML 的客户端的应用，设计相关的实验，帮助用户了解 XML 以及相关技术的应用，并为 XML 在后续服务器端的应用做一个铺垫。

5.1　预 备 知 识

5.1.1　XML 标记语言基础

XML 是可扩展标记语言(extensible markup language)的简称，是一种半结构化的语言。一方面，XML 语言与 SGML(standard generalized markup language)语言一致，只定义数据内容，不具有任何处理数据的过程。处理和显示 XML 数据是通过其他技术来实现的。这样的一种定义形式，导致了 XML 语言具有扩展性、灵活性、自描述性和简单性，适应当前网络应用的要求。因此，XML 得到了广泛的支持和应用。

1. XML 文件结构

XML 文件书写形式简单，XML 文件由数字、任何字符构成，任何文本编辑器就可以定义，只要保存文件为.XML 或.xml 后缀即可。一般来说，一个 XML 文件从逻辑结构可以分成处理指令、文件声明、标记、实体引用、注释和 CDATA 片段 6 个部分。

(1) 处理指令。XML 文件的处理指令必须放置在 XML 头部，向应用程序传递的特殊指令。处理指令形如“＜?处理指令?＞”，指令常见下列两种情况。

① ＜?xml?＞处理指令。表示定义一个 XML 文件，必须放置在 XML 文件的第一行的位置，通知应用程序对该文件按照 XML 文件处理，该处理指令包含的常见属性见表 5-1。

表 5-1　＜?xml?＞指令的常见属性

属　性	类别	说　　明
version	必选	说明 XML 的版本
encoding	可选	指定 XML 采用的字符集
standalone	可选	指定 XML 是否可使用外部的 DTD

② ＜?xml-stylesheet?＞处理指令。表示按照指定的样式表处理 XML 文件。该指令具有常见的属性见表 5-2。

(2) 文件声明。XML 文件是通过 DOCTYPE 声明文件的使用空间，通过内部定义或引入外部的文件(如 DTD 文档类型定义文件)，限定 XML 文件使用的文法。DOCTYPE 往往要说明根元素和文档。具体的定义形式有如下。

表 5-2　<?xml-stylesheet?>的常见属性

属　性	类别	说　　明
type	必选	指定样式表的类型，例如可以是 text/css
href	必选	指定要显示处理的样式表文件
title	可选	值为字符串，表示链接文档指定的标题
charset	可选	定义目标 URL 的字符编码方式
alternate	可选	说明样式表是否是可替换的样式表。如果值为 yes，则样式表只能在用户要求才能加载；如果值为 no 表示样式表为默认加载的样式表
media	可选	说明样式表适用的媒体环境，取值见表 5-3

表 5-3　<?xml-stylesheet?>指令的 media 属性的取值

值	说　　明	值	说　　明
aural	声音屏幕	braille	盲文屏幕
handheld	小型设备	print	纸张
projection	投影到大屏幕	tty	定宽的终端
tv	WebTV	screen	显示器

① 内部声明，用 DOCTYPE 元素中直接内嵌文件声明，形如：

```
<!DOCTYPE 根元素 [
  内部声明
]>
```

② 结合内部和外部声明，即可用 DOCTYPE 元素引入说明文件类型的外部文件，也可以在声明内部直接指定文件声明，有下列两种形式。

```
<!DOCTYPE 根元素 SYSTEM"外部 DTD 文件"[
  内部声明
]>
```

或

```
<!DOCTYPE 根元素 PUBLIC"公共标记" "外部文件 URL"[
  内部文件声明
]>
```

(3) 标记。XML 不像 XHTML 一样有预先定义的标记，XML 的标记是由用户自定义产生的。但是即便如此，XML 的标记必须遵循一定的规则。

① XML 的标记区分大小写。

② 标记组成 XML 元素，XML 元素必须用标记进行封闭。非空的元素是用成对的形如"<标记>PCDATA 段</标记>"；空元素必须用空格和"/"说明，形如<标记/>。

③ XML 文件中必须有一个根元素。

④ XML 标记可有属性，但属性值必须通过双引号包含起来。

⑤ XML 元素必须正确的嵌套。元素必须被其父元素完全包含。

XML 文件实际上就是一个倒置的 XML 树。如果某元素包含其他元素，则某元素就

是其他元素的父结点，而其他元素就是某元素的子结点。因此，根元素为所有元素的父结点。

（4）实体引用。实体引用就是具有特定含义的字符对象。实体引用可以用两种形式实现。& 字符串和 &＃实体编号。

（5）注释。注释是在 XML 文件中提供解释说明。形如＜！－－说明－－＞。在应用程序使用 XML 文件时，注释会被忽略。

（6）CDATA 片段。PCDATA 片段是包含在元素内部的内容，会被应用程序解析。而 CDATA 片段是不会被应用程序解析处理的片段，CDATA 片段会在解析后按原样完整保留。通常 CDATA 用于一些特殊符号正常显示。CDATA 片段的定义形式如下：

```
<![CDATA[
文本片段
]]>
```

2. XML 的命名空间

为了预防不同文件的同名标记产生冲突，XML 文件中可以用命名空间对这些同名标记进行区分。XML 的命名空间是一组关于元素和属性命名唯一的集合的名称。XML 的命名空间的定义形式如下：

```
<命名空间前缀：元素名 xmlns：命名空间前缀="命名空间 URI">
```

在定义命名空间后，可以直接使用命名空间的前缀来表示命名空间标识符。

3. XML 的数据岛

XML 数据岛是嵌入到 HTML 中的 XML 数据，能被微软的 Internet Explorer 5.x 版本及以上版本识别。在 HTML 中嵌入 XML 数据岛的形式如下：

```
<xml id="数据岛名" src="XML 文件 URL">
</xml>
```

或

```
<xml id="数据岛名" src=" XML 文件 URL"/>
```

定义数据岛的目的是使用 XML 数据，这就需要将 XML 数据绑定到具体的 HTML 中。通常 HTML 的标记 table 可用属性 datasrc 绑定到具体的 XML 数据岛，然后可以使用标记 span 或标记 div 的属性 datafld 来引用具体的 XML 数据岛。形式如下：

```
<table datasrc="#数据岛名">
  <tr>
    <td>
      <span datafld="xml 标记"/>
    </td>
      ⋮
</tr>
      ⋮
</table>
```

或

```
<table datasrc="#数据岛名">
  <tr>
    <td>
      <div datafld="xml 标记"/>
    </td>
    ⋮
</tr>
    ⋮
</table>
```

5.1.2 XML 的相关技术

XML 文件本身只是作为数据存储的作用，它必须和其他技术结合来实现特定的应用。图 5-1 展现了与 XML 相关的技术。

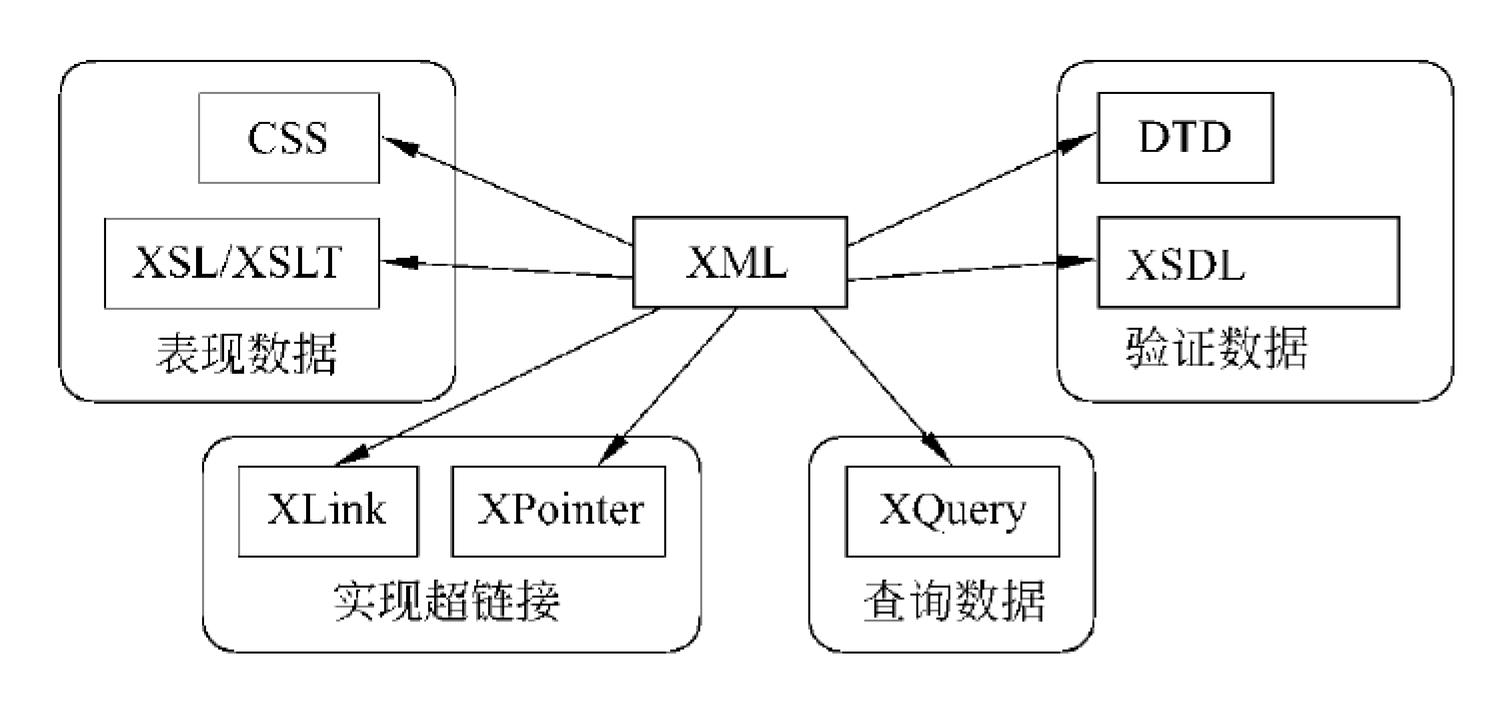


图 5-1　XML 相关技术示意

1. CSS 层叠样式表

CSS 可以和 XML 直接结合，可在 XML 文件中用"<? xml-stylesheet? >"处理指令，指定引用的样式表文件，将 XML 数据按照定义的样式显示。CSS 样式定义说明见本书第 3 章。

2. XSL/XSLT

XSL 是扩展样式表语言。XSL 由 3 个部分组成：XSLT(扩展样式表转换)、XPath 与 XSL-FO。

(1) XPath。XPath(XML Path Language)是 XML 路径语言，用于 XML 导航，实现对 XML 数据的定位。它是实现 XSLT、XPointer 以及解析 XML 的前提基础。XPath 采用目录结构的路径表达方式来对 XML 数据定位，并提供了内置的标准函数，为准确表达 XML 数据提供方便。XPath 的具体应用见配套教材 5.4 节。

(2) XSLT。XSLT 与 CSS 一样，可表示 XML 数据。但不同之处在于，XSLT 将 XML 数据转换成其他格式，而 CSS 只是定义 XML 数据的表示样式，不会修改 XML 数据。XSLT 转换的原理如图 5-2 所示。

(3) XSL-FO。XSL-FO(XML Style Language Fomatting Objects)用于格式化 XML 数据的语言。



图 5-2　XSLT 的工作原理

3. DTD

DTD(Document Type Definition)是文档类型定义,说明 XML 的文档结构,常用来验证 XML 文件的合法性。XML 文件中可以使用 DOCTYPE 引入外部的 DTD 文件或直接定义内部的 DTD 来实现对 XML 文件的验证。

4. XSDL

XSDL(XML Schema Definition Language)表示 XML 模式定义语言,也称 XSD。用于定义 XML 结构,是验证 XML 数据的另一种方式。与 DTD 相比,XSDL 基于 XML,可以表达元素的属性的数据类型,具有扩展、灵活、简单、易学等特性,被认为是 DTD 的替代者。

5. XLink 和 XPointer

XLink(XML Linking Language)表示 XML 链接语言,定义创建超链接的标准方法。XPointer(XML Pointer Language)表示 XML 指针语言,实现超链接指向 XML 内容。通常 XLink 和 XPointer(XML Pointer Language)结合使用,实现对访问资源的操作。

6. XQuery

XQuery(XML Query Language)是 XML 查询语言,用于查找和提取 XML 元素和属性。XQuery 往往用于网络数据的提取。

5.1.3　JavaScript 访问 XML 数据

在第 4 章中了解了 JavaScript 的应用。JavaScript 也可以与 XML 结合,开发具有实际意义的应用。这是因为,大部分浏览器中具有内置的 XML 解析器,而这些解析器将 XML 数据加载到内存中,将 XML 数据转换成为 JavaScript 可处理访问的对象。JavaScript 可根据实际需要对 XML 数据进行处理。通过这样的一个过程,使得 JavaScript 访问 XML 数据成为可能。

1. 加载 XML 数据

由于用户采用的浏览器不同,导致浏览器内置的 XML 解析器也会有所不同。从而使得浏览解析器在加载 XML 数据的形式上也有所不同。

(1) 微软 Internet Explorer 加载 XML 数据。

```
var xmlDoc=new ActiveXObject("Microsoft.XMLDOM");
//创建一个空的 XML 文档对象
xmlDoc.async=false;
//关闭异步加载
xmlDoc.load("XML 文件名.xml");
//通知解析器加载 XML 文档
```

如果需要加载的是 XML 数据的文本片段，则将上述的最后一行代码改写成下列形式：

```
xmlDoc.loadXML("xml 字符串名");          //通知解析器加载 XML 字符串
```

（2）其他浏览器加载 XML 数据。

```
xmlDoc=document.implementation.createDocument("","",null);
//创建一个空 XML 文档对象
xmlDoc.async="false";                    //关闭异步加载
xmlDoc.load("XML 文件名.xml");           //通知解析器加载 XML 文档
```

不同于微软 Internet Explorer 使用 loadXML()解析 XML 字符串，其他浏览器如 FireFox 等通过 DOMParser 实现对 XML 字符串的加载，形式如下：

```
var parser=new DOMParser();
//创建空的 XML 文档对象
var doc=parser.parseFromString(XML 字符串名,"text/xml");
//加载 XML 字符串
```

注意：关闭异步加载的一个用处是保证文档在完全加载之前，JavaScript 脚本不会继续执行。

2. 解析 XML 数据

加载 XML 文件后，就可以通过创建的 XML DOM(XML 文档对象)来实现对 XML 数据的解析，使得 JavaScript 理解 XML 数据，达到处理 XML 数据的目的。XML DOM 就是提供了一套解析 XML 数据的方法，具体内容见表 5-4。

表 5-4　XML DOM 的常见方法

方　法	说　明	方　法	说　明
getElementsByTagName(name)	根据标记名访问结点	replaceChild(node)	用新结点替换当前结点
appendChild(node)	添加新结点	createElement(element)	创建新结点
removeChild(node)	删除结点	createTextNode(node)	创建新的文本结点

在解析过程中将 XML 数据视之 XML 树，XML 树中的每一个成分为结点，结点类型见表 5-5。然后按照结点类型中的相应方法执行对结点的处理(如修改、插入和删除等)。

表 5-5　XML DOM 的结点

结点类型	说　明	结点类型	说　明
Document	表示 XML 整个文档	Attr	表示元素的属性
DocumentFragment	表示部分 XML 文档	Text	表示元素的文本内容
DocumentType	表示文档类型	CDATASecetion	表示 CDATA 片段
ProcessingInstruction	表示处理指令	Comment	表示注释
EntityReference	表示扩展实体引用	Entity	表示实体
Element	表示 XML 元素	Notation	表示 DTD 声明的符号

5.2　实验 5.1　XML 的验证机制

实验目的：

(1) 了解 XML 的文件结构以及语法特性，并能熟练编制 XML 文件。

(2) 熟练掌握 DTD 文档类型定义的基本要求，并使用 DTD 定义 XML 文档结构。

(3) 熟练掌握和应用 XSDL 语言的基本内容，并使用 XSDL 定义 XML 文档结构。

(4) 比较 DTD 和 XSDL 两种 XML 的验证机制。

实验内容：

本实验是设计保存通讯录数据的 XML 文件。在这个实验中，要求创建一个表示通讯录的 XML 文件。分别用 DTD 和 XSDL 定义该 XML 文件的文件结构，并分别将定义的 DTD 文件和 XSD 文件导入到表示通讯录的 XML 文件进行验证，从而达到了解 XML 文件的定义、XML 的两种验证机制的目的。

实验步骤：

练习 1：设计和建立 XML 文件。

本次练习的目的是了解 XML 文件结构。要求如下：

(1) 选定并打开一个文本编辑器(如 Dreamweaver CS 3.0)，也可以选择 XMLSpy 专业软件，然后新建一个 XML 类型的文件。

(2) 打开给定 ContactList.xml 文件，代码见程序清单 5-1。该文件中定义通讯录的每条记录可包括数据内容有记录的编号、姓名、关系(朋友、同事、亲属、一般)、工作单位、职位、电话、手机、电子邮件。该文件存在 5 处错误，请指出错误，并修改。

程序清单 5-1：

```
<!--ContactList.xml-->
<? xml version="1.0" encoding="ISO-8859-1"? >
<记录 编号="AS00001">
  <姓名>陈易</姓名>
  <关系>朋友</关系>
  <工作单位>江南乐海客车厂</工作单位>
  <职位>
  <电话>0791-5454543</电话>
  <手机>1309999999</手机>
  <电子邮件>chenyi@163.com</电子邮件>
</记录>
<记录>
  <编号>AS00002</编号>
  <姓名>黄明</姓名>
  <关系>同事</关系>
  <工作单位>江南师范附小</工作单位>
```

```
    <职位>教师</职位>
    <电话>010-32323244<手机>13093564349</电话></手机>
    <电子邮件>huangming@163.com</电子邮件>
</记录>
```

练习 2：用 DTD 定义 XML 文件结构。

本次练习是针对修改后的 ContactList.xml，打开以有 ContactList.dtd 文件（代码见程序清单 5-2），用于定义对应的 XML 文件结构。具体要求如下：

（1）修改程序清单 5-2，使得每条记录的记录的编号（具有唯一性）、姓名为必选，手机和电话数据是至少有一个选中，其他数据内容要求可选。

程序清单 5-2：

```
<!--ContactList.dtd-->
<!ELEMENT 通讯录 ((记录+))>
<!ELEMENT 记录 ((姓名,关系, 工作单位, 职位, 电话|手机,电子邮件))>
<!ATTLIST 记录
编号 CDATA #REQUIRED
>
<!ELEMENT 姓名 (#PCDATA) #REQUIRED>
<!ELEMENT 关系 (#PCDATA) #IMPLIED>
<!ELEMENT 工作单位 (#PCDATA) #IMPLIED>
<!ELEMENT 职位 (#PCDATA) #IMPLIED>
<!ELEMENT 电话 (#PCDATA) #IMPLIED>
<!ELEMENT 手机 (#PCDATA) #IMPLIED>
<!ELEMENT 电子邮件 (#PCDATA) #IMPLIED>
```

思考：定义的 DTD 文件是否可以实现练习的要求？如果没有实现是什么原因？如果实现了，说明 DTD 的语言特性。

（2）如果为 ContactList.xml 的每条记录增加一个“来源”属性，该属性的值只能为“江蝉的通讯录”，形如“<记录 编号="00001" 来源="江蝉的通讯录">…</记录>”，请修改已定义的 DTD 文件，使得对应的 XML 可以通过验证。

（3）假设为 ContactList.xml 的记录增加一个空元素“记录结束”，请修改 ContactList.dtd，使之满足要求。

练习 3：用 XSDL 定义 XML 文件结构。

本次练习是用 XSDL 重新定义 ContactList.xml 的文件结构。文件 ContactList.xsd（代码见程序清单 5-3），是用 XSDL 语言描述的 ContactList.xml 文件结构。该 xsd 文件指定通讯录的记录有：姓名、关系（朋友、同事、亲属、一般）、工作单位、职位、电话、手机、电子邮件。请将 代码 1 ～ 代码 8 补充完整。

程序清单 5-3：

```
<?xml version="1.0" encoding="UTF-8"?>
<!--ContactList.xsd-->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="通讯录">
```

```
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="记录" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>

<xs:element name="记录">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="姓名"/>
            <xs:element ref="关系"/>
            <xs:element ref="工作单位"/>
            <xs:element ref="职位"/>
            <xs:element ref="电话"/>
            <xs:element ref="手机"/>
            <xs:element ref="电子邮件"/>
        </xs:sequence>
        <xs:attribute name="编号" use="required">
            代码 1
            <!--说明编号属性的类型-->
        </xs:attribute>
    </xs:complexType>
</xs:element>

<xs:element name="姓名">
    代码 2
    <!--补充姓名的类型-->
</xs:element>

<xs:element name="关系">
    代码 3
    <!--补充关系的类型-->
</xs:element>

<xs:element name="工作单位">
    代码 4
    <!--补充工作单位的类型-->
</xs:element>

<xs:element name="职位">
    代码 5
    <!--补充职位的类型-->
</xs:element>
```

```
    <xs:element name="电话">
        代码 6
        <!--补充电话的类型-->
    </xs:element>

    <xs:element name="手机">
        代码 7
        <!--补充手机的类型-->
    </xs:element>

    <xs:element name="电子邮件">
        代码 8
        <!--补充电子邮件的类型-->
    </xs:element>
</xs:schema>
```

(1) 按照下列要求,将程序清单 5-3 补充完整。

① 记录中说明编号属性、编号属性的类型必须为整数。

② 记录中的姓名、工作单位和职位指定的数据类型为字符串。

③ 记录中的关系必须指定为朋友、同事、亲属、一般中的一种,不能重复。

④ 记录中的电话数据必须按照形式"区号-号码"表示,其中区号占 3 位,号码占 8 位。

⑤ 记录中的手机数据必须为数字类型,长度介于 7～12 位之间。

⑥ 记录中的电子邮件的数据的格式形如"邮件名@域名"。

(2) 如果将上述的 ContactList.xml 文件中,为记录元素增加一个空元素"记录结束",则需要对对应的 XSD 文件做怎样修改? 比较 XSD 的非空元素、带属性的元素、嵌套子元素的元素以及空元素的定义,并说明用 XSD 定义这些元素的特点。

(3) 比较 DTD 与 XSDL 的语言特色,并写出二者的区别,并用本实验的练习进行举例说明。

(4) 请用户参考本书或其他教材的目录结构,用 XML 定义选中书籍的目录,并用 XSDL 语言对定义的 XML 文件进行验证。

5.3 实验 5.2 显示 XML 数据

实验目的:

(1) 掌握和运用 XML 数据岛解决问题。

(2) 掌握和熟练运用 CSS 处理 XML 数据显示。

(3) 理解和掌握 XPath 技术规范。

(4) 理解和掌握 XSLT 技术规范,并运用 XSLT 转换 XML 数据,实现显示 XML。

(5) 初步了解 JavaScript 结合 XSLT 转换 XML 的过程。

实验内容：

设计和实现一个显示电影列表的网页，要求该电影列表中定义若干电影信息，每个电影信息中记录了电影的名称、电影制作地区、电影的导演、主要演员、电影类型、上映时间及海报。其中，电影导演、主要演员和电影类型可以是多项内容。

实验步骤：

本次实验由 4 个相关的练习构成。按照练习提供的步骤依次完成任务。本次实验的电影列表用 XML 定义在 FilmList. xml 文件中，具体代码见程序清单 5-4。

程序清单 5-4：

```
<?xml version="1.0" encoding="UTF-8"?>
<!--FilmList.xml-->
<FilmList>
    <Film id="00001">
        <Name>赤壁</Name>                              <!--电影名称-->
        <Place>中国</Place>                            <!--地区-->
        <Director>吴宇森</Director>                    <!--导演-->
        <Star>梁朝伟、金城武、张震、张丰毅</Star>      <!--主要演员-->
        <Type>战争片、历史片</Type>                    <!--类型-->
        <Schedule>2008 年 7 月 1 日</Schedule>         <!--上映时间-->
        <Description>三国时期，刘备和孙权两支大军在赤壁与曹操的部队大战，用火攻打败曹军
的历史故事。</Description>                            <!--简介-->
        <Poster>image/cibi.jpg</Poster>
    </Film>
    <Film id="00002">
        <Name>功夫熊猫</Name>
        <Place>欧美</Place>
        <Director>Mark Osborme、John StEvEnson</Director>
        <Star>Jack Black、成龙、Dustin Hoffman</Star>
        <Type>动画片、动作片、喜剧片</Type>
        <Schedule>2008 年 6 月 6 日</Schedule>
        <Description>熊猫阿宝习练中国功夫，打败凶猛邪恶的雪豹泰龙的故事。</Description>
        <Poster>image/panda.jpg</Poster>
    </Film>
    <Film>
        <Name>地狱男爵 2：黄金部队</Name>
        <Place>欧美</Place>
        <Director>Guillermo DelToro</Director>
        <Star>Selma Blair、RonPerIman、Jeffrey Tambor</Star>
        <Type>恐怖片、科幻片、动作片</Type>
        <Schedule>2008 年 5 月 19 日</Schedule>
```

```
        <Description>“地狱男爵”和奇幻世界的弩阿达王子斗争的故事。</Description>
        <Poster>image/hellboy.jpg</Poster>
    </Film>
</FilmList>
```

练习 1：用数据岛绑定 XML 数据。

本次练习是通过用数据岛实现对 XML 数据的显示。要求：

(1) 设计和实现用数据岛显示 XML 数据。要求：打开 Internet Explorer 5.x 以上版本的浏览器，运行网页 FilmList.html（代码见程序清单 5-5），观察并记录运行结果。修改该网页的代码，使得该网页中用表格绑定 FilmList.xml 文件，并将该 XML 文件中的信息按照表格形式显示输出。表格每一行有一部电影的信息，海报放置在每一行的第一个单元格，其他信息可以明显区别，运行结果类似图 5-3。



图 5-3 数据岛绑定 XML 数据

程序清单 5-5：

```
<!--FilmList.html-->
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>数据岛的应用</title>
</head>
<body>
<xml id="filmlist" src="FilmList.xml"/>
<table width="690">
<caption>新片上映</caption>
<tr>
```

```
    <th width="50"> </th>
    <th width="53">名称</th>
    <th width="50">地区</th>
    <th width="79">导演</th>
    <th width="171">主要演员</th>
    <th width="38">类型</th>
    <th width="137">上映时间</th>
    <th width="76">简介</th>
  </tr>
  </table>
  <table  bgcolor="#999999" border="1" bordercolor="#333333" >
  <tr>
      <td>
      <img datafld="Poster" width="60" height="80"/>
  <!--海报-->
      <span datafld="Name"/>
  <!--电影名称-->
      <span datafld="Place"/>
  <!--地区-->
      <span datafld="Director"/>
  <!--导演-->
      <span datafld="Star"/>
  <!--主要演员-->
      <span datafld="Type"/>
  <!--电影类型-->
      <span datafld="Schedule"/>
  <!--电影上映时间-->
      <span datafld="Description"/>
  <!--电影简介-->
  </td>
    </tr>
  </table>
  </body>
  </html>
```

(2) 在实现上述功能的 FilmList.html 中继续修改,使得网页中的记录可以分页显示。要求:每个页面中可以显示两条电影信息记录,定义 4 个按钮分别具有单击显示第一页的记录、上一个的记录、下一个记录和最后一个记录的功能。选择"第一页"按钮和选择"下一页"按钮的运行结果如图 5-4 和图 5-5 所示。

(3) 将上述定义功能完整的 FilmList.html 网页在 DreamWeaver CS(或 XMLSpy)中验证,记录验证结果。然后在其他浏览器(如 FireFox)中打开,观察并记录运行结果。解释运行结果产生的原因?

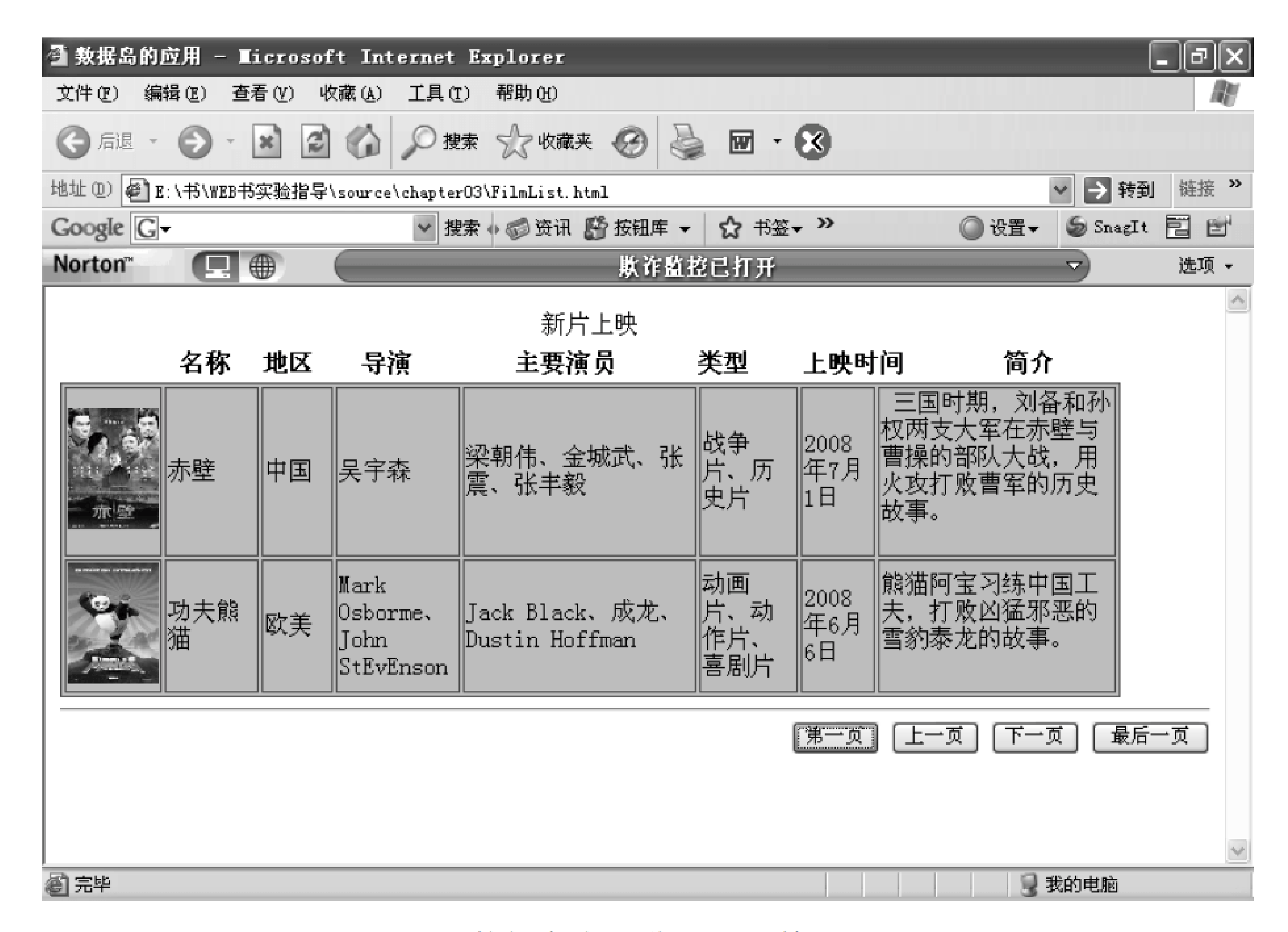


图 5-4　数据岛实现分页显示第一面记录



图 5-5　数据岛实现分页显示下一面记录

练习 2：CSS 显示 XML 数据。

本次练习是为 FilmList.xml 文件用 CSS 定义显示样式。要求如下：

（1）阅读外部 CSS 样式表 FilmList.css 文件，具体内容见程序清单 5-6，请重新定义该样式表内容，使得 FilmList.xml 文件的显示效果为表格，表格的每一行记录一个电影信息。其中每一个电影记录中，电影名称和其他信息分成两列，对于表格单元格的记录要求前景颜色和背景颜色不同（颜色可以自选），使得每一行的记录可以明显区分。

程序清单 5-6：

```
/* FilmList.css */
FilmList{                          /* 定义 Film List 标记的样式 */
        background-color:#000000;
        width:800;
        height:600;
}
Film{                              /* 定义 Film 标记的样式 */
       color:#3A2885;
       border:#333333;
       }
Name{                              /* 定义 Name 标记的样式 */
       background-color:#3A2885;
       color:#00A6AD;
       font:x-large;
       }
Place,Star,Schedule{               /* 定义 Place,Star,Schedule 标记的样式 */
              background-color:#00A6AD;
       }
Director,type,Description{         /* 定义 Director,type,Description 标记的样式 */
       background-color:#205AA7;
       color:#999999;
       }
```

（2）在 FilmList. xml 中加入指令

```
<?xml-stylesheet type="text/css" href="FilmList.css"?>
```

使得 FilmList. xml 的运行结果类似图 5-6。

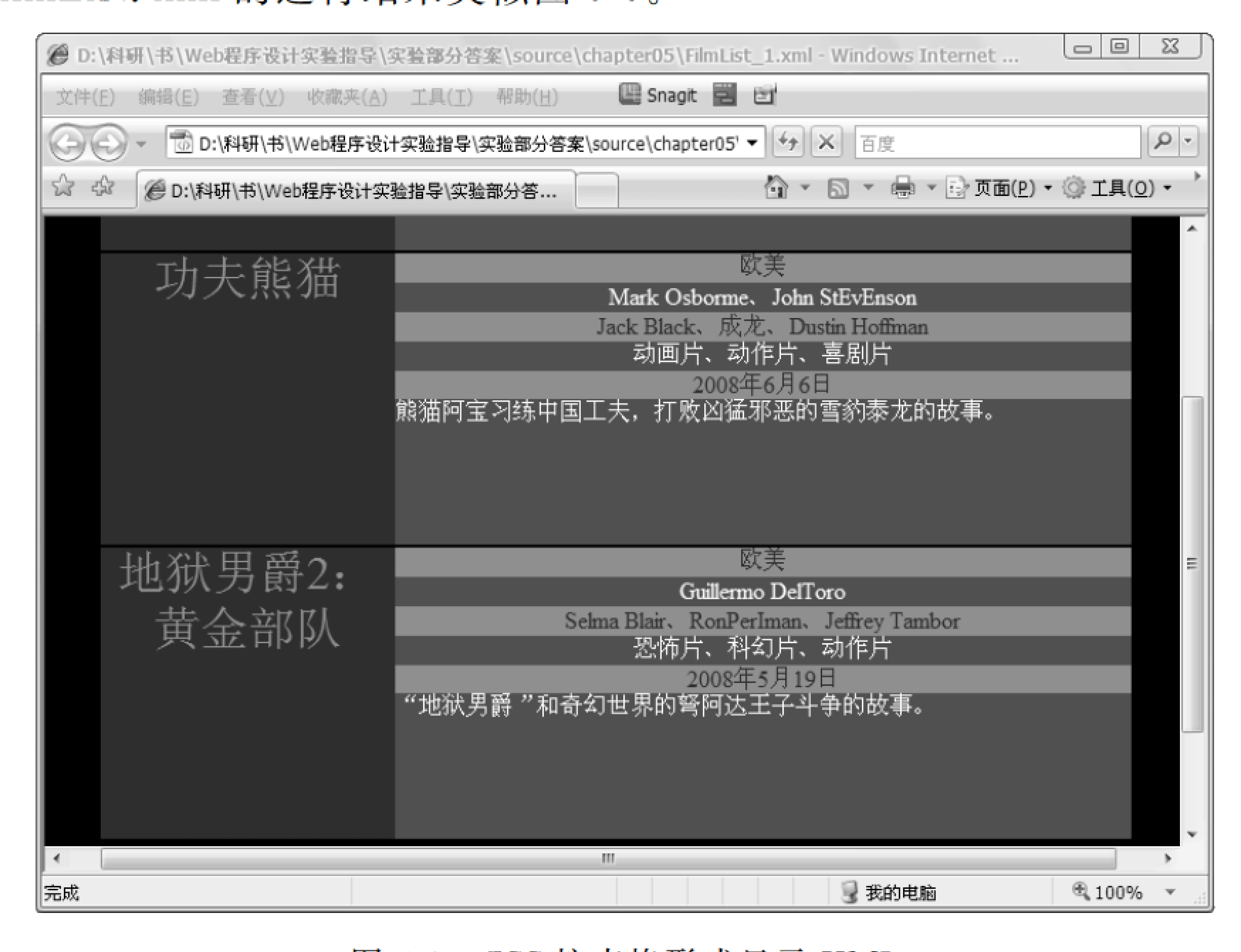


图 5-6　CSS 按表格形式显示 XML

思考：

(1) 在图 5-6 中没有显示电影的海报图片，请问有什么办法可以让 XML 文件显示海报图片？请写出代码，并在浏览器中运行你的解决方法。如果无法显示 XML 文件中的不同的海报图片，请写出原因。

(2) 比较 CSS 显示 XML 数据和上述用数据岛显示 XML 数据的异同。请写出二者的不同之处。

练习 3：XSLT 显示 XML 数据。

本次练习用 XSLT 转换 XML 数据，达到表现 XML 数据的效果。在本次练习中仍采用上述的 FilmList. xml 文件。然后按照下列的步骤依次完成练习。具体内容如下：

(1) 下列是用 XSL 定义的一个 XSLT 文件 FilmList. xslt，希望能显示电影列表中所有的记录，具体内容见程序清单 5-7。然后修改 FilmList. xml，取消使用 FilmList. css 定义的 CSS 样式，而是声明使用 FilmList. xslt 定义的 XSLT 样式表，具体内容见程序清单 5-8。在浏览器中运行 FilmList. xml，观察运行结果是否与图 5-7 一致。如果运行结果与希望的结果不一致，请解释原因。

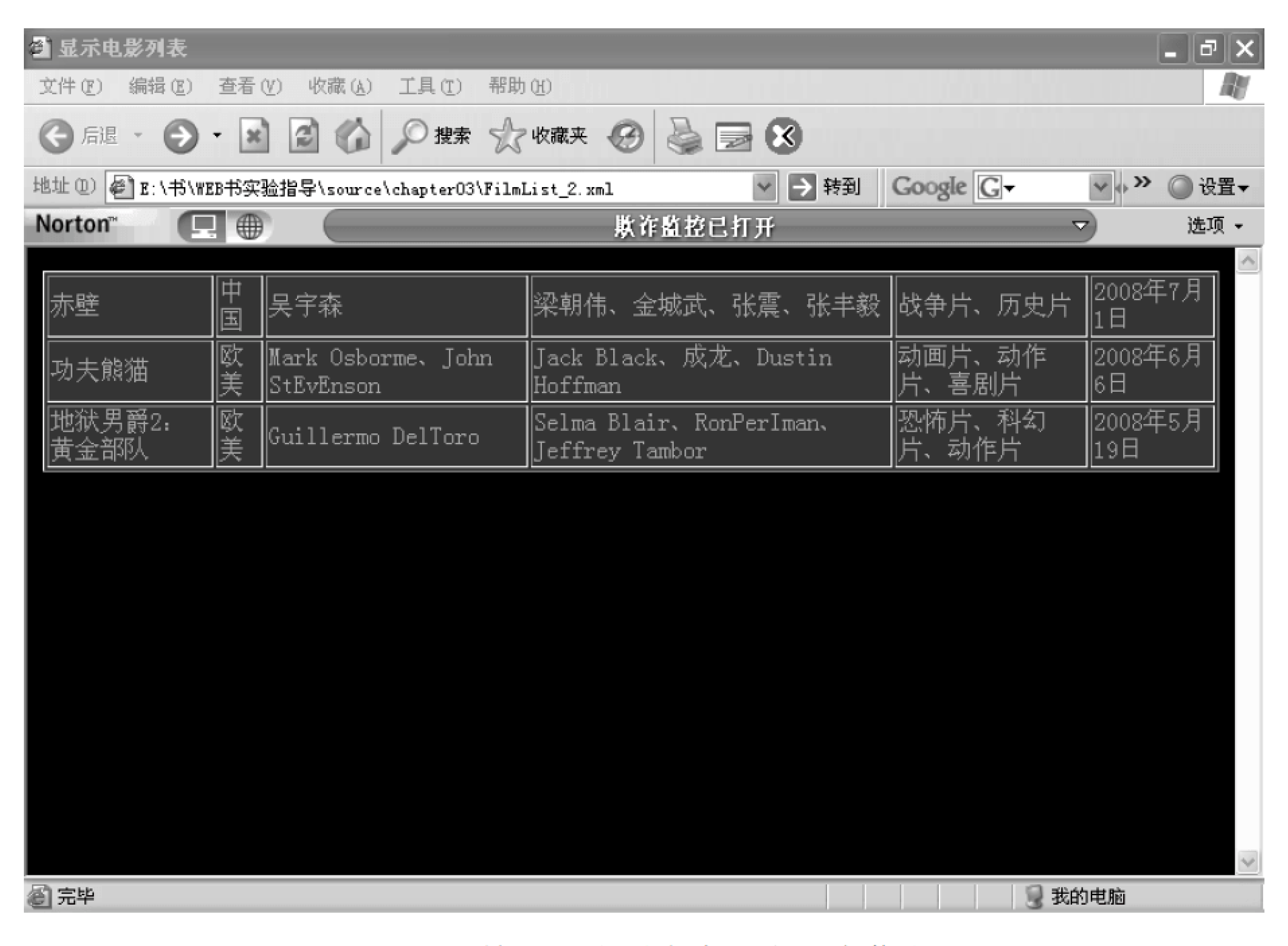


图 5-7 输出电影列表中所有文本信息

程序清单 5-7：

```
<?xml version="1.0" encoding="GB2312"?>
<xsl:stylesheet version="2.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:fn="http://www.w3.org/2005/xpath-functions">
<xsl:output method="xml" version="1.0" encoding="GB2312" indent="yes"/>
<!--FilmList.xslt-->
```

```
<xsl:template match="/">
<html>
        <head>
            <title>显示电影列表</title>
        </head>
        <body bgcolor="#000000">
          <table bgcolor="#333333" border="1">
            <xsl:for-each select="FilmList">
             <tr style="color:#CCCCCC" bgcolor="#333333">
               <td><xsl:value-of select="Name"  /></td>
               <td><xsl:value-of select="Place"  /></td>
               <td><xsl:value-of select="Director"  /></td>
               <td><xsl:value-of select="Star"  /></td>
               <td><xsl:value-of select="Type"  /></td>
               <td><xsl:value-of select="Schedule"  /></td>
             </tr>
            </xsl:for-each>
         </table>
        </body>
</html>
</xsl:template>
</xsl:stylesheet>
```

程序清单 5-8:

```
<?xml version="1.0" encoding="GB2312"?>
<?xml-stylesheet type="text/xsl" href="FilmList.xslt"?>
<!--FilmList.xml 声明使用 XSLT 文件-->
<FilmList>
    <Film>
        <Name>赤壁</Name>
        <Place>中国</Place>
        <Director>吴宇森</Director>
        <Star>梁朝伟、金城武、张震、张丰毅</Star>
        <Type>战争片、历史片</Type>
        <Schedule>2008 年 7 月 1 日</Schedule>
        <Description>三国时期,刘备和孙权两支大军在赤壁与曹操的部队大战,用火攻打败曹军
的历史故事。
    </Description>
        <Poster>image/cibi.jpg</Poster>
    </Film>
    <Film>
        <Name>功夫熊猫</Name>
        <Place>欧美</Place>
        <Director>Mark Osborme、John StEvEnson</Director>
```

```
        <Star>Jack Black、成龙、Dustin Hoffman</Star>
        <Type>动画片、动作片、喜剧片</Type>
        <Schedule>2008 年 6 月 6 日</Schedule>
        <Description>熊猫阿宝习练中国功夫，打败凶猛邪恶的雪豹泰龙的故事。
    </Description>
        <Poster>image/panda.jpg</Poster>
    </Film>
    <Film>
        <Name>地狱男爵 2：黄金部队</Name>
        <Place>欧美</Place>
        <Director>Guillermo DelToro</Director>
        <Star>Selma Blair、RonPerIman、Jeffrey Tambor</Star>
        <Type>恐怖片、科幻片、动作片</Type>
        <Schedule>2008 年 5 月 19 日</Schedule>
        <Description>"地狱男爵"和奇幻世界的弩阿达王子斗争的故事。
        </Description>
        <Poster>image/hellboy.jpg</Poster>
    </Film>
</FilmList>
```

（2）修改 FilmList.xslt 文件，要求显示电影列表中的所有电影信息（包括海报图片），具体显示格式的要求如下：每条电影记录中的"海报"缩略图片（宽：90px；高：120px）和其他文字信息分两列显示；"海报"缩略图是一个图片链接。即单击"海报" 缩略图片，可导航到"海报"的原尺寸大小的图片；放置文本信息的列由 6 行构成，不同行的背景色和前景色不同，行和行可以明显区分。运行结果如图 5-8 所示。



图 5-8　按 XSLT 定义样式输出

(3) 修改上述 FilmList.xslt,要求只显示电影列表中属于动作片类型的电影记录。显示样式同上,运行结果如图 5-9。思考:要求显示上映时间在 2008 年 6 月份以后的电影记录,或要求显示电影的 id 为奇数的电影记录之类,或要求按照演员名字进行排序等问题,该如何处理,请写出你的处理办法。



图 5-9 显示条件记录信息

(3) 利用 XML 和 XSLT 自行设计和实现一个关于介绍电影的网站,或主题另选。

练习 4:XSLT+JavaScript 转换 XML 数据。

由于存在有些浏览器不支持 XML 文件的直接显示,所以需要将 XML 文件转换成其他格式的文件。本次练习的目的是了解在客户端用 XSLT 转换 XML 文件成为其他形式的文件如 HTML、XHTML 等,主要内容是用 XML+XSLT+JavaScript 等技术,设计一个提供候车时刻表信息的服务网站。按照下列步骤依次完成练习。

(1) 已知用 XML 定义火车时刻表在 TrainList.xml 中,具体代码见程序清单 5-9。然后用 XSLT 为该 XML 文件定义的样式文件 TrainList.xslt,具体内容见程序清单 5-10,样式要求每一条记录在表格中的每一行中显示。最后运行程序清单 5-11 中定义的 XHTML 文件 TrainList.html,运行 TrainList.html。观察运行结果。解释说明程序清单 5-11 中的 JavaScript 定义的脚本中的每一行功能,并修改该脚本,使之能在其他浏览器中也可以运行。

程序清单 5-9:

```
<?xml version="1.0" encoding="gb2312"?>
<!--TrainList.xml-->
<TrainList>
<Train>
   <Line>Z17</Line>                                   <!--车次-->
```

```
    <Type>直达特快</Type>                          <!--类型-->
    <From>北京西</From>                            <!--始发站-->
    <To>长沙</To>                                  <!--终点站-->
    <GoOff>18:00</GoOff>                           <!--出发时间-->
    <ArrivalTime>07:21</ArrivalTime>               <!--达到时间-->
    <Distance>1587 公里</Distance>                 <!--里程-->
</Train>
<Train>
     <Line>K7</Line>                               <!--车次-->
    <Type>空调普快</Type>                          <!--类型-->
    <From>汉口</From>                              <!--始发站-->
    <To>广州</To>                                  <!--终点站-->
    <GoOff>19:28</GoOff>                           <!--出发时间-->
    <ArrivalTime>08:46</ArrivalTime>               <!--达到时间-->
    <Distance>1089 公里</Distance>
</Train>
…<!--略,其他候车时刻请自行补充-->
</TrainList>
```

程序清单 5-10:

```
<?xml version="1.0" encoding="GB2312"?>
<!--TrainList.xslt-->
<xsl:stylesheet version="2.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:fn="http://www.w3.org/2005/xpath-functions">
<xsl:output method="xml" version="1.0" encoding="GB2312" indent="yes"/>
<xsl:template match="/">
<html xmlns="http://www.w3.org/1999/xhtml">
      <head>
        <title>火车时刻列表</title>
      </head>
      <body>
       <table>
              <caption>火车时刻表</caption>
              <tr>
                  <th>车次</th>
                  <th>始发站</th>
                  <th>终点站</th>
                  <th>发车时间</th>
                  <th>达到时间</th>
                  <th>里程</th>
              </tr>
              <xsl:for-each select="TrainList/Train">
```

```
                <tr>
                    <td><xsl:value-of select="Line"/></td>
                    <td><xsl:value-of select="From"/></td>
                    <td><xsl:value-of select="To"/></td>
                    <td><xsl:value-of select="GoOff"/></td>
                    <td><xsl:value-of select="ArrivalTime"/></td>
                    <td><xsl:value-of select="Distance"/></td>
                </tr>
                </xsl:for-each>
        </table>
       </body>
</html>
</xsl:template>
</xsl:stylesheet>
```

程序清单 5-11：

```
<!--TrainList.html-->
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
<body>

<script type="text/javascript">
// 装载 TrainList.xml 文件
var xml=new ActiveXObject("Microsoft.XMLDOM")
xml.async=false
xml.load("TrainList.xml")

//装载 TrainList.xslt 文件
var xsl=new ActiveXObject("Microsoft.XMLDOM")
xsl.async=false
xsl.load("TrainList.xslt")

// XSLT 转换 XML,并输出转换结果
document.write(xml.transformNode(xsl))
</script>
</body>
</html>
```

（2）由于火车时刻表的记录非常多，请用定义 JavaScript 定义脚本，实现火车时刻表分页显示，每一页 6 个记录，并通过“第一页”、“上一页”、“下一页”和“最后一页”这 4 个按钮，帮助用户翻页浏览，运行结果如图 5-10 所示。代码见程序清单 5-12，请按照要求，将该程序从 代码 1 ～ 代码 5 处补充完整。



线路	类型	起始站	终点站	出发时间	达到时间	里程
Z17	直达特快	北京西	长沙	18:00	次日：07:21	1587公里
Z18	直达特快	长沙	北京西	18:00	次日：07:11	1587公里
K7	空调普快	汉口	广州	19:28	次日：08:46	1089公里
K8	空调普快	广州	汉口	16:28	次日：05:46	1089公里
T88	空调特快	青岛	北京西	18:00	次日：07:11	1587公里
T17	空调特快	北京西	长沙	18:00	次日：07:11	1587公里

图 5-10　分页显示示意图

程序清单 5-12：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<!--TrainList1.html-->
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>火车列表</title>
</head>

<body>
<script language="javascript">
var pageNum=6;
//每页的记录数
var page=0;
//前一页面
var Text="";
//需要显示的内容;
var xmlDoc =new ActiveXObject("Microsoft.XMLDOM");
xmlDoc.async="false" ;
代码 1 ;
//补充代码,实现加载 TrainList.xml
header="<table width='80%'>"+
    "<caption>火车列表</caption>"+
    "<tr><th>线路</th>"+
      "<th>类型</th>"+
      "<th>起始站</th>"+
      "<th>终点站</th>"+
      "<th>出发时间</th>"+
```

```
        "<th>达到时间</th>"+
        "<th>里程</th></tr>";

    var maxNum=xmlDoc.getElementsByTagName("Train").length;
    //检索的记录数
    var column=xmlDoc.getElementsByTagName("Train").item(0).childNodes;
    //记录的列数
    var colNum=column.length;
    var pagesNumber=Math.ceil(maxNum/pageNum)-1;
    //页数
function firstPage()
    //第一页{
  代码 2 ;
//补充代码,显示第一页
}

function GoFirstPage(){
代码 3 ;
//补充代码,转向第一页
}

function lastPage(){
//最后页面
    thePage="<input type='button' value='最后一页' onClick='JavaScript:return GoLastPage
()'/>";
        return thePage;
}

function GoLastPage(){
//转向最后页
   page=pagesNumber;
   getContent();
   Text="";
}

function UpPage(page) {
//上一个页面
   代码 4 ;
//补充代码,实现显示上一个页面按钮
    return thePage;
}
function NextPage(page)
//下一页
{
    var thePage="<input type='button' value='后一页' disabled='disabled'/>";
    if(page<pagesNumber)
```

```
    thePage="<input type='button' value='后一页' onClick='JavaScript:return GoNextPage()
'/>";
        return thePage;
}

function GoUpPage(){
//前一页
  代码 5 ;
//补充代码,转向上一个页面
    getContent();
    Text="";
}

function GoNextPage()
//后一页
{
if (page<pagesNumber) page++;
    getContent();
    Text="";
}

function pageInfo(page)
//显示分页信息
{
    var pb;
    pb=firstPage()+" "+UpPage(page)+"  "+NextPage(page)+"  "+lastPage();
    return pb;
}

function getContent(){
    if (!page) page=0;
    n=page * pageNum;
    endNum=(page+1) * pageNum;
    if (endNum>maxNum) endNum=maxNum;
    Text=header+Text;
    for (;n<endNum;n++){
        Text=Text+"<tr>";
         for (m=0;m<=6;m++){
          mName=column.item(m).tagName;
          Text=Text+("<td>"+
                xmlDoc.getElementsByTagName(mName).item(n).text+
            "</td>");
          }
          Text=Text+"</tr>"
          mm="";
          }
          resultHtml.innerHTML=Text+"</table>"+pageInfo(page);
```

```
        Text=""
}
</script>
<div id="resultHtml">
<script>
if (maxNum==0)
    {
        document.write("没有火车记录")
    }
    else{
        getContent();
    }
</script>
</div>
</body>
</html>
```

思考：请结合 XML、XSLT 和 JavaScript 实现上述同样的功能。其中，XML 实现数据定义，XSLT 实现数据转换，JavaScript 实现为数据的有条件选取提供条件。

第 6 章　WAP 2.0 编程

伴随移动设备用户的日益增多，以及无线 Internet 的广泛应用和普及，有必要了解无线应用的相关技术。WAP 是开发无线应用的佼佼者。一方面，WAP 得到广大的无线移动设备的支持；另一方面，WAP 为无线移动设备提供访问互联网的内容和高级数据。因此，WAP 是沟通无线应用和互联网的桥梁。本章将从开发无线网页的角度出发，着重介绍 WAP 2.0 的 WML、XHTML MP、WMLScript、WCSS。

6.1　预 备 知 识

6.1.1　WAP 2.0 概述

WAP 表示无线应用协议。当前的最新版本是 WAP 2.0。与 WAP 1.x 相比，WAP 2.0 具有新的特点。

(1) WAP 2.0 采用双协议架构。WAP 2.0 不但兼容支持 WAP 1.x 的 WSP/WTP/WDP 协议，而且支持 HTTP、TCP/IP、TLS 协议。这样使得 WAP 2.0 的网关不必向 WAP 1.x 的网关一样，要将 WSP/WTP/WDP 转换成 HTTP/TLS/TCP 协议，如图 6-1 所示，而是直接通过 HTTP/TCP 访问，提高了运行速度和执行性能，见图 6-2。

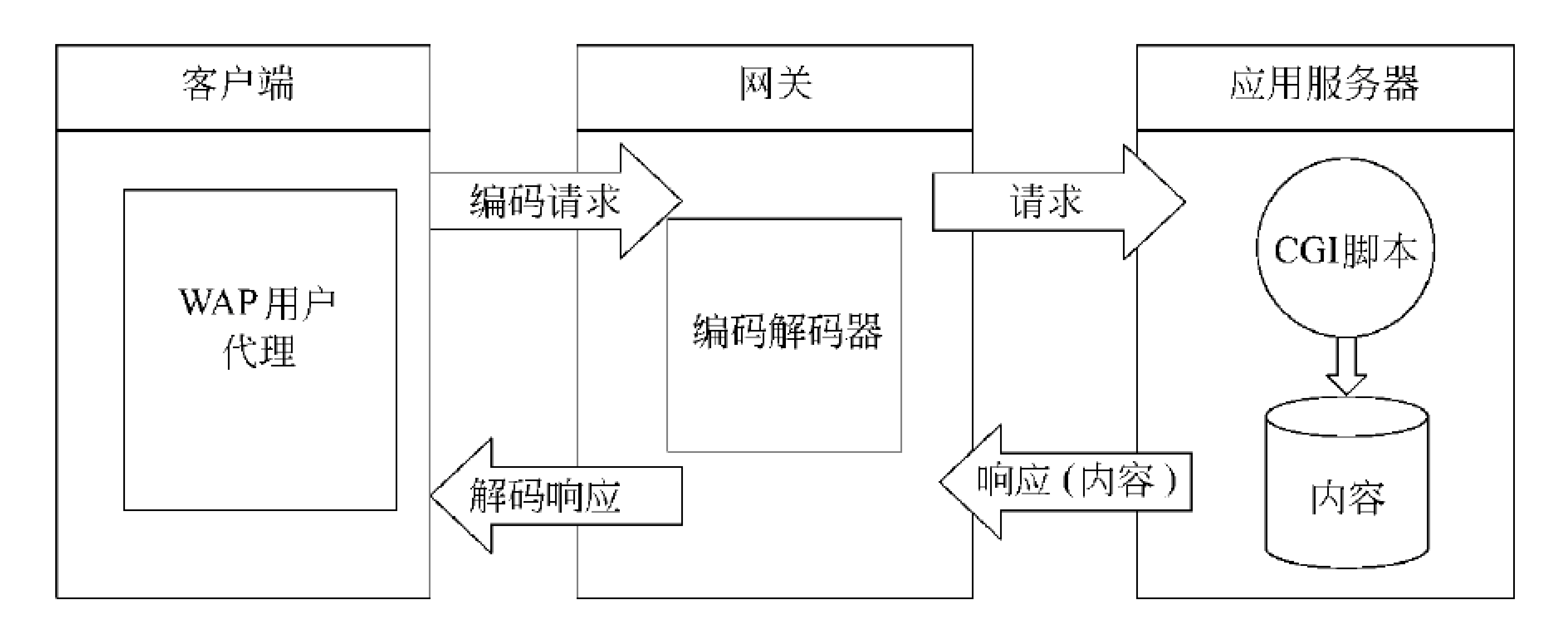


图 6-1　WAP 1.x 的编程模型

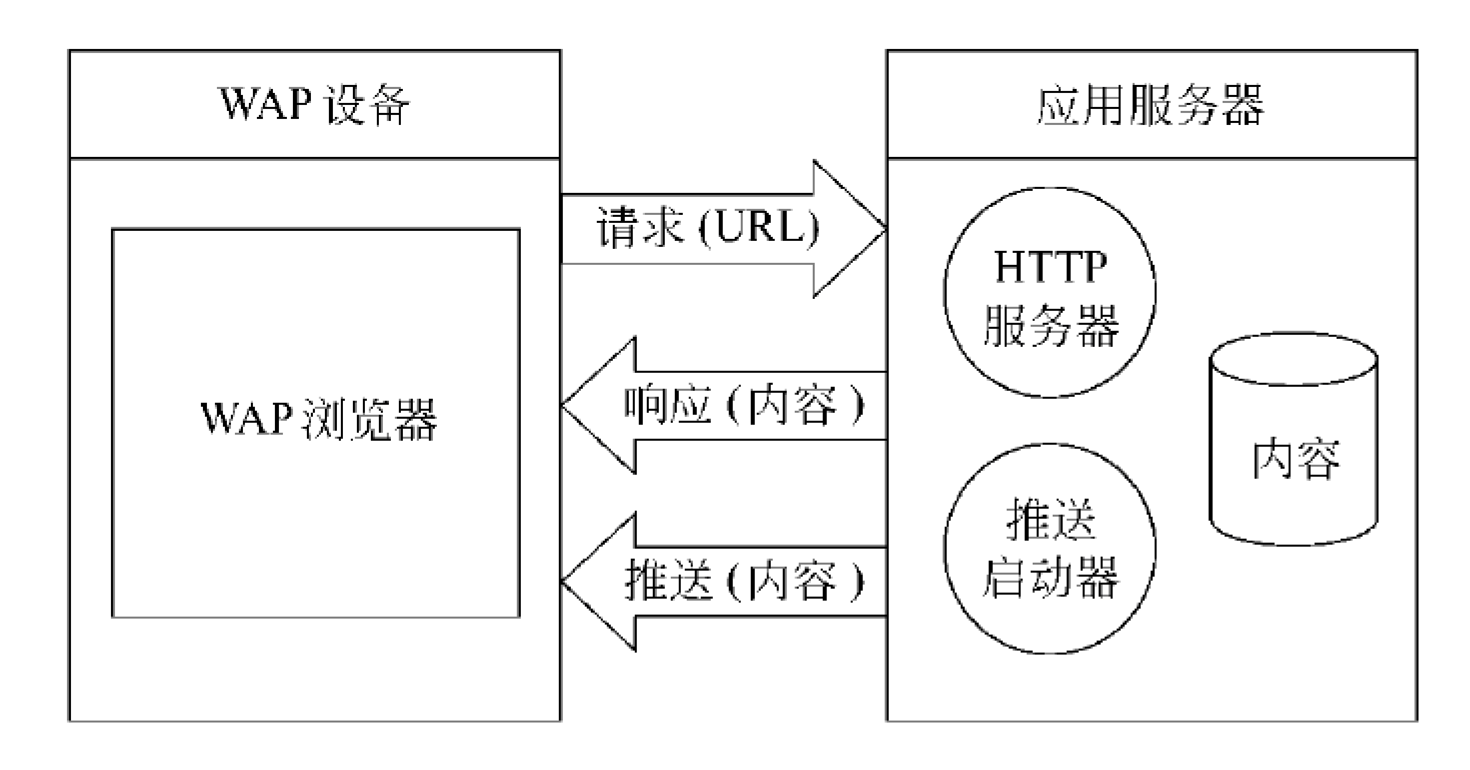


图 6-2　WAP 2.0 的编程模型

(2) WAP 2.0 突破了网络环境的制约。无论在 2.5G 通信网络还是 3G 通信网络,无线移动设备可以通过 WAP 2.0 访问 Internet。

(3) WAP 2.0 提供了丰富的应用环境。不但兼容并继续支持了 WAP 1.x 的 WML 1.x,而且还支持 XHTML MP、WML 2.0、WCSS 等。使得无线应用环境进一步和 Internet 的应用环境趋向一致。

(4) WAP 2.0 可以进一步降低无线移动设备消耗的电力,使各种无线移动设备使用时间更为长久。

6.1.2 WAP 2.0 支持的标记语言

WAP 2.0 定义的无线应用环境中允许使用的标记语言有 WML 1.x、WML 2.0、XHTML MP。其中,WML 1.x 是 WAP 2.0 兼容 WAP 1.x 而来,而 WML 2.0 和 XHTML MP 是 WAP 2.0 新增的标记语言。

1. WML 1.x

WML 1.x 是无线标记语言,是 XML 1.0 的子集,与 XML 的语法规则保持一致。WML 1.x 定义了一套实现包括文本、图像、音频等内容描述标记库,通过标记库设计和实现 WAP 网页。

2. WML 2.0

WML 2.0 也是一种无线标记语言,虽然名称与 WML 1.x 一致,但是表示的内容发生了巨大的变化。这是因为 WML 2.0 是接受 XHTML Basic 规范的产物,是 XHTML MP 的超集。一方面,凡是可以用 XHTML Basic 定义的内容一律采用 XHTML Basic 和 WCSS 来实现;另一方面,对于不可用 XHTML Basic 和 WCSS 表示的内容,仍可以用 WML 1.x 的语义表示。为了与 WML 1.x 区别,WML 2.0 在原 WML 1.x 语义中增加了前缀 wml:。

3. XHTML MP

XHTML MP 表示扩展超文本链接语言移动描述,是 XHTML 1.0 的子集,具有 XHTML 所有的语法要求。XHTML MP 具有 XHTML Basic 的所有内容,但并不与 WML 1.x 兼容。此外,XHTML MP 不再支持锚点,不支持显示下划线的 u 标记,不支持软键盘,不支持脚本,不支持变量,不支持事件,不支持 timer 定时器,不支持 input 输入框的 format 属性等内容。通常 XHTML MP 与 WCSS 结合使用。

表 6-1 WML 1.x、WML 2.0 和 XHTML MP 的常见标记

类 别	WML 1.x	WML 2.0	XHTML MP
文件结构	wml、card、head、access 和 template	body、head、html、title、wml:card 和 wml:access	body、head、html 和 title
文本显示及布局	b、big、em、i、strong、small、up、pre 和 br	abbr、acronym、address、blockquote、br、cite、code、dfn、div、em、strong、var、h1、h2、h3、h4、h5、h6、kbd、p、pre、q、samp、span、b、big、small 和 u	abbr、acronym、address、blockquote、br、cite、code、dfn、div、em、strong、var、h1、h2、h3、h4、h5、h6、kbd、p、pre、q、samp、span、b、big 和 small
超链接	a 和 anchor	a、link 和 wml:anchor	a 和 link
水平分割		hr	hr
列表		dd、dl、dt、li、ul 和 ol	dd、dl、dt、li、ul 和 ol

类　别	WML 1.x	WML 2.0	XHTML MP
表单	select、optgroup、option、input 和 fieldset	form、input、label、select、option、textarea、fieldset 和 optgroup	form、input、label、select、option、textarea、fieldset 和 optgroup
表格	table、tr 和 td	table、td、th、tr 和 caption	table、td、th、tr 和 caption
图像	img	img	img
对象		object 和 param	object 和 param
基址		base	base
链接		link	link
元信息	meta	meta	meta
样式模块		style	style
任务	go、pre、refresh 和 noop	wml：go、wml：pre、wml：refresh 和 wml：noop	
事件	do、onevent 和 postfield	wml：do、wml：onevent 和 wml：postfield	
变量	setvar 和 timer	wml：getvar、wml：setvar 和 wml：timer	

6.1.3　WCSS

WCSS(Wireless Profile Cascading Style Sheet，无线描述层叠样式表)，是 CSS 2.0 的子集和 WAP 的特别扩展构成。WCSS 必须和 XHTML MP 结合使用。由于不同移动设备具有不同的物理特性，例如屏幕的大小、分辨率不同。而 WCSS 的出现，为不同的移动设备处理显示样式提供了可能，充分体现了当前 Internet 世界中数据内容和数据表现相分离的原则，比 WML 具有更好的控制表现的功能。

但是，WCSS 也有自身的一些问题。在第一次使用 WCSS 样式时，必须将全部的 WCSS 样式文件从服务器下载到手机中，一方面，增加了服务器的工作负担；另一方面，在一定程度上影响了 WAP 网页的执行效率。

WCSS 作为 CSS 2.0 的子集，具有与 CSS 2.0 一致的语法规则。

(1) WCSS 的样式声明形式如下：

```
选择器 1[,选择器 2,…,选择器 m]{
        属性名 1：属性值；
        [属性名 2：属性值；
            ⋮
        属性名 n：属性值]
}
```

选择器指定需要定义样式的标记，选择器可以有一个，也可以有多个。选择器之间用逗号(，)分隔。选择器有“派生选择器”、“类选择器”、“ID 选择器”、“全局选择器”、“组选择器”等。这些选择器的类型与 CSS 2.0 定义的类型一致。属性名是指希望修改属性的名称，取值由属性值指定。属性的定义也可以有多个，属性定义之间用分号(；)分开。常见的 WCSS 的属性及说明见表 6-2，其中“跑马灯”、“软键盘”和“输入”是 WAP 扩展属性，具体见表 6-2。

表 6-2　WCSS 的常见属性

类型	属　　性	说　　明	类型	属　　性	说　　明
框模型	border-width	边框宽度	字体	font-family	字体
	border-top-width	上边框宽度		font-style	字体样式
	border-right-width	右边框宽度		font-variant	字体的异体
	border-bottom-width	下边框宽度		font-weight	字体的粗细
	border-left-width	左边框宽度		font-size	字体的尺寸
	border-color	边框颜色		font	字体
	border-top-color	上边框颜色	列表	list-style-type	列表样式类别
	border-right-color	右边框颜色		list-style-position	列表样式定位
	border-left-color	左边框颜色		list-style-image	列表样式图片
	border-bottom-color	下边框颜色		list-style	列表样式
	border-style	边框样式	文本	text-indent	文本缩进
	border-top-style	上边框样式		text-align	对齐文本
	border-right-style	右边框样式		text-decoration	文本修饰
	border-left-style	左边框样式		text-transform	文本变形
	border-bottom-style	下边框样式		white-space	空白
	border-top	上边框		color	设置文本颜色
	border-right	右边框	显示格式化	display	设置显示文本效果
	border-left	左边框		float	相对显示内容的位置
	border-bottom	下边框		clear	清除
	padding	内边距		width	宽度
	padding-top	上内边距		height	高度
	padding-right	右内边距		vertical-align	水平排列
	padding-bottom	下内边距	可视	visibility	可视性
	padding-left	左内边距	跑马灯	display	设置显示效果(值为-wap-marquee)
	margin	外边距			
	margin-top	上外边距		-wap-marquee-style	跑马灯样式(值有slide/scroll/alternate)
	margin-right	右外边距			
	margin-bottom	下外边距		-wap-marquee-loop	跑马灯循环
	margin-left	左外边距		-wap-marquee-dir	跑马灯方向(值为 ltr/rtl)
颜色和背景	background-color	背景颜色		-wap-marquee-speed	跑马灯的速度(值有fast/normal/slow)
	background-image	背景图片			
	background-repeat	反复背景图	输入	-wap-input-format	设置输入格式
	background-attachment	放置背景			
	background-position	背景定位		-wap-input-required	输入留白(值为 true/false)
	background	背景	软键盘	-wap-accesskey	设置软键盘

（2）WCSS 中可以用“/ * 注释 * /”来说明注释，达到解释代码的作用。注释可以在多行中显示。在浏览器解释 WAP 网页时，WCSS 的注释会被忽略。

（3）WCSS 的 MIME 的类型仍是 text/css，定义的 WCSS 样式文件后缀仍为 css。

6.1.4 WMLScript

WMLScript 是一种服务于移动设备的脚本语言，是基于 ECMAScript 脚本语言发展起来的。它往往和 WML 1.x 结合开发无线应用，服务于 WML 1.x 语言。WMLScript 并不是嵌入到 WML 网页内部，而是一个独立的外部文件，该外部文件在服务器端的 WAP 网关编译成二进制文件，然后下载到移动设备中，为相关的 WML 网页提供相应的支持。WMLScript 文件有两种形式。

（1）扩展名为 wmls，是文本文件，保存 WMLScript 的源程序代码。

（2）扩展名为 wmlsc，是二进制的字节码文件，也就是源程序文件编译后的文件。

WMLScript 的主要优点如下：

（1）用于用户输入的合法性检验。

（2）扩展移动设备的功能。

（3）提供友好的人机界面设计，为良好的人机交互提供可能。

（4）克服移动服务网络的带宽的限制，提供丰富的程序功能。所以，有必要了解 WMLScript 脚本语言。

6.1.5 WAP 网页设计要点

与第 2 章介绍的网页设计原则类似，设计一个应用在无线移动设备上的 WAP 网页也需要考虑用户对象、WAP 网页的主题、WAP 网页的色彩体系等要求。除此之外，开发 WAP 网页还要充分考虑移动设备是性能受限的设备，表现在显示分辨率、色彩、运算能力、存储能力、电池消耗都有所不同。更重要的是，移动设备具有的功能是不一致的。这就决定了 WAP 网页在设计上具有自身的特点。

（1）建立有效合理的 WAP 的导航模型，用户可以快速学会和使用导航。对于实现导航的“选择列表”、“复选框”等多种组件，应考虑输入形式最为简单的导航方式，快速导航到指定的位置。其次，导航层次不超过 5 层，不宜过深，否则用户容易失去导航目标。这意味着，用 WML 1.x 开发网页，WML 内不能超过 5 张 card。

（2）设计时，要充分考虑目标设备的屏幕大小。WAP 网页的内容不能过多，言简意赅。WAP 网页的文本最好居中显示，使用的样式前后保持一致。不要使用过多的色彩，影响用户阅读。

（3）要合理使用图片和表格，如果图片不是主要内容，要慎用。对于屏幕过小的移动设备，利用表格分开内容没有太大的意义，所以也要慎用。

（4）如果必须要用图片，图片的格式必须适用于最大范围的移动设备。WAP 网页使用的图片数量不能太多。另外，每幅图片也不能太大，否则会影响执行效率。在使用图片中要确定图片的宽度和高度。

（5）设计有效的反馈信息，对于用户操作的结果，最好给出反馈，引导用户作出正确的操作。

（6）充分考虑目标设备的存储能力，WAP 网页不能过大。这表示网页文件中的注释要简单明了，不能过多占据文件大小。尽量少用 class 和 id 的应用样式。

（7）用 XHTML MP 开发网页，网页使用的字符集为 Unicode，确保网页文件可以让 WAP 浏览器正确解析。

6.2 实验 6.1 开发和测试 WAP 应用的工具

实验目的：

（1）了解常见开发 WAP 网页的开发工具。

（2）了解测试 WAP 网页的主要模拟器。

（3）初步了解简单的 WAP 网页的制作和调试。

实验内容：

（1）在 Windows 环境下，安装 WAP 网页的编辑器 Nokia Mobile Internet ToolKit 4.1。

（2）安装和使用 WinWap for Windows 4.1 浏览器。

（3）安装和使用 OpenWave 6.22 浏览器。

实验步骤：

练习 1：安装 Nokia Mobile Internet Toolkit 4.1。

Nokia Mobile Internet Toolkit 4.1 是诺基亚公司提供的免费开发工具软件。要正确安装该软件，需要到 http://www.forum.nokia.com/网站下载本软件，本软件的 Patch for JRE 5.0 的补丁文件以及请求软件的序列码。

（1）首先，必须安装 Sun 公司开发的 JDK 5.0 以上版本的 Java 开发工具包。如果已安装可以直接进入第(2)步。

（2）安装 Nokia Mobile Internet Toolkit 4.1 编辑器。

① 打开安装软件 NMIT_41.exe，会出现一个启动画面，如图 6-3 所示。

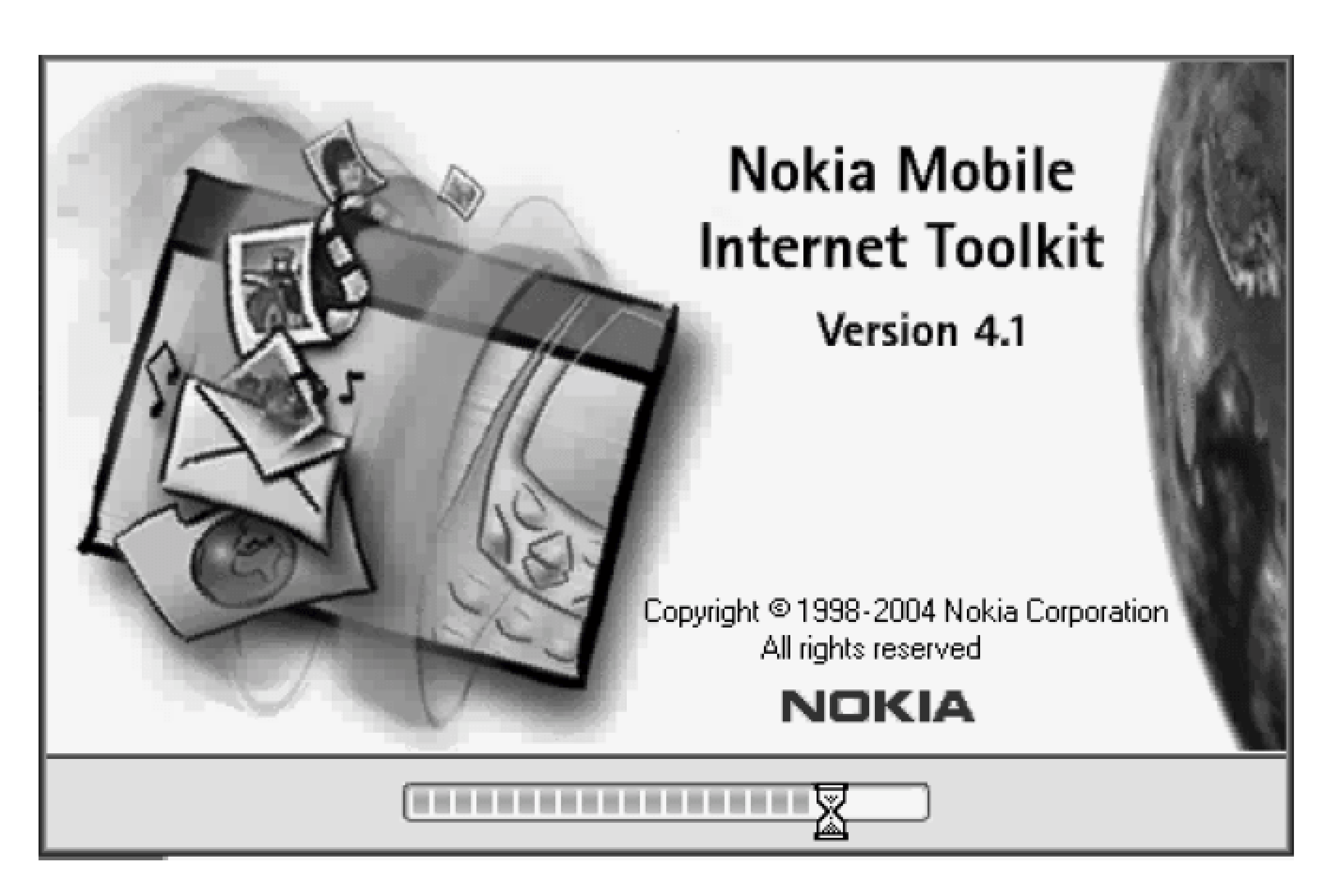


图 6-3　Nokia Mobile Internet Toolkit 的启动

② 当启动完毕，进入安装软件的“平台需求”画面，如图 6-4 所示。单击 Next 按钮，出现“介绍”界面，如图 6-5 所示。单击 Next 按钮继续安装。

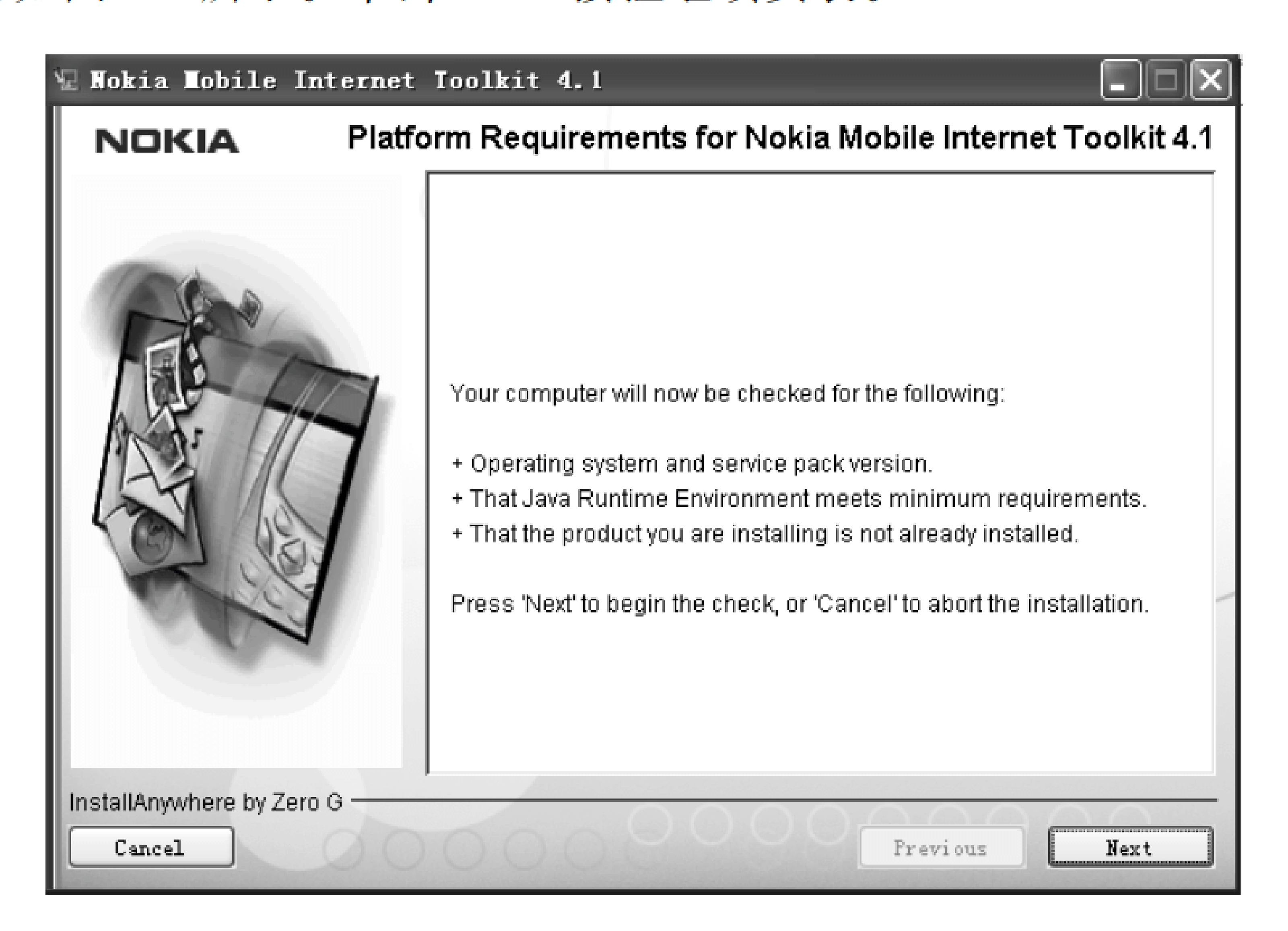


图 6-4　平台需求界面

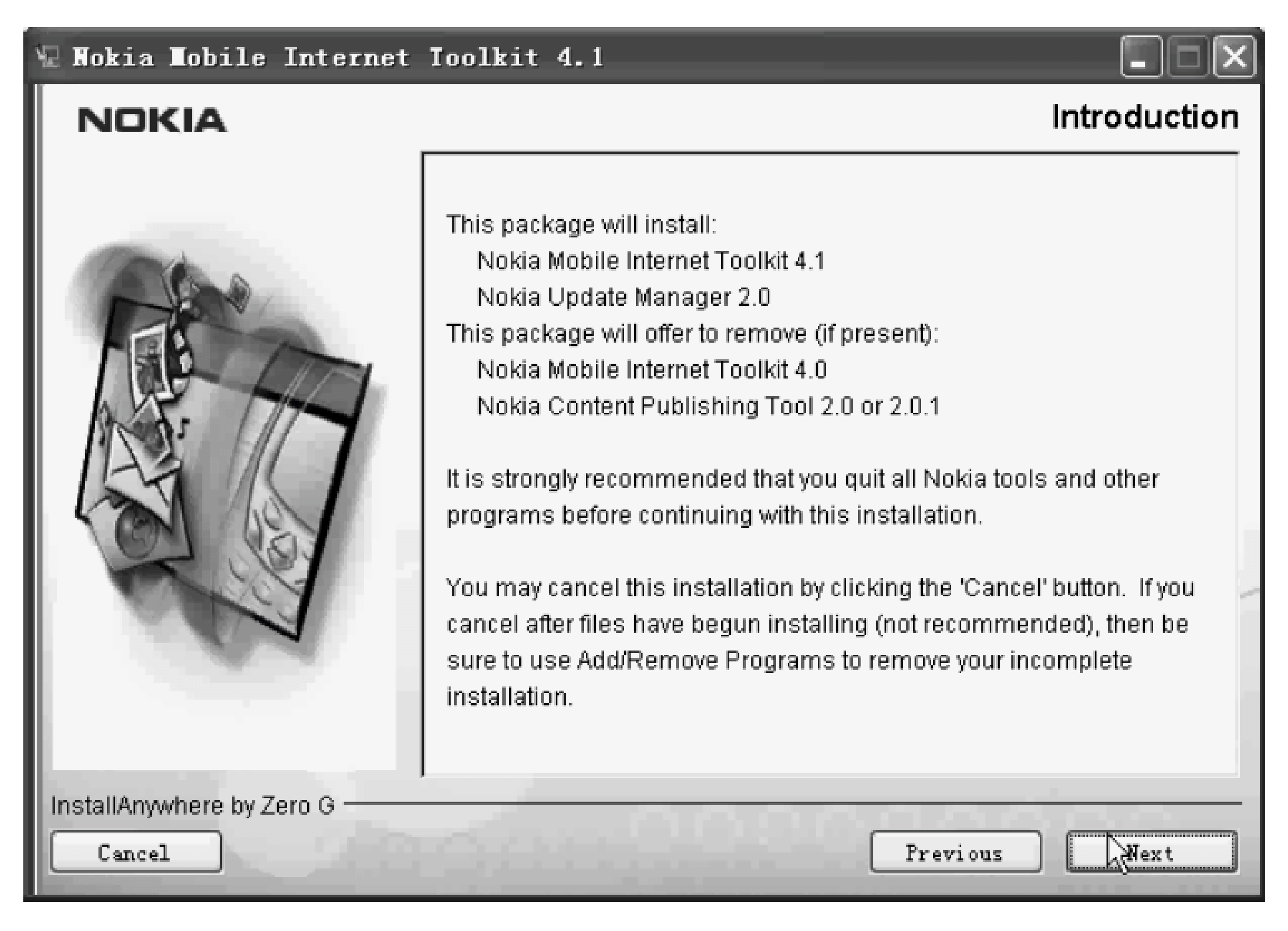


图 6-5　介绍界面

③ 在如图 6-6 所示“用户软件许可协议”界面。单击 I accept the terms of the License Agreement 单选按钮。

④ 在如图 6-7 界面中，按照要求输入 Nokia Forum 网站的用户名和请求的序列码后，选择 Next 按钮。其中，用户名和序列码可以到 Nokia Forum 网站免费申请获得。

⑤ 再在设定目录界面中设定安装软件的根目录，如图 6-8 所示。可以单击 Choose 按钮重新设置根目录。

⑥ 设置根目录后，在随后出现的“关联文件类型”界面中设置 Nokia Mobile Internet Toolkit 关联支持的文件类型，如图 6-9 所示。也可以选择 All of the above 复选框表示选择所有的文件类型。然后选择 Next 按钮进入下一个步骤。

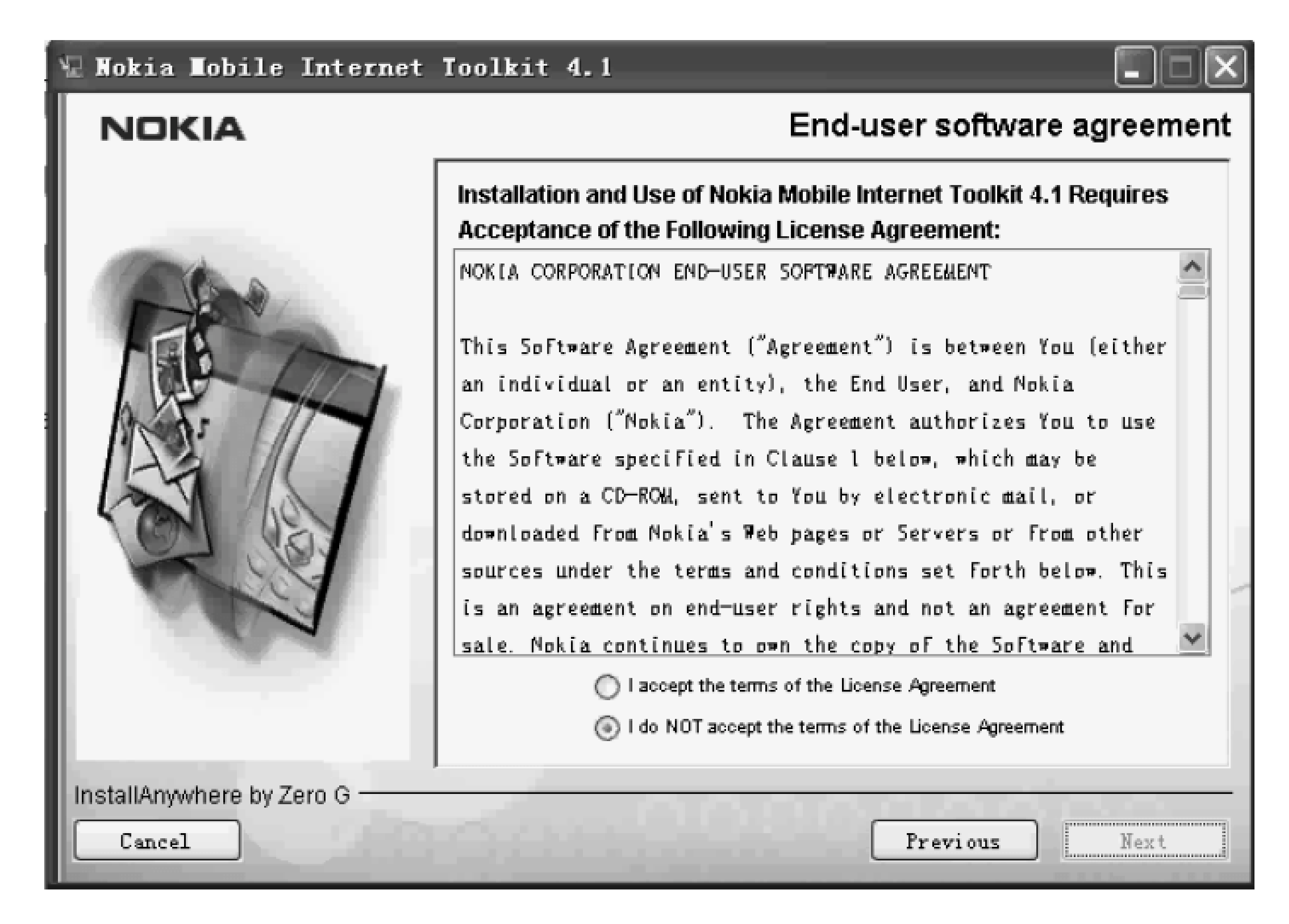


图 6-6　用户许可协议界面

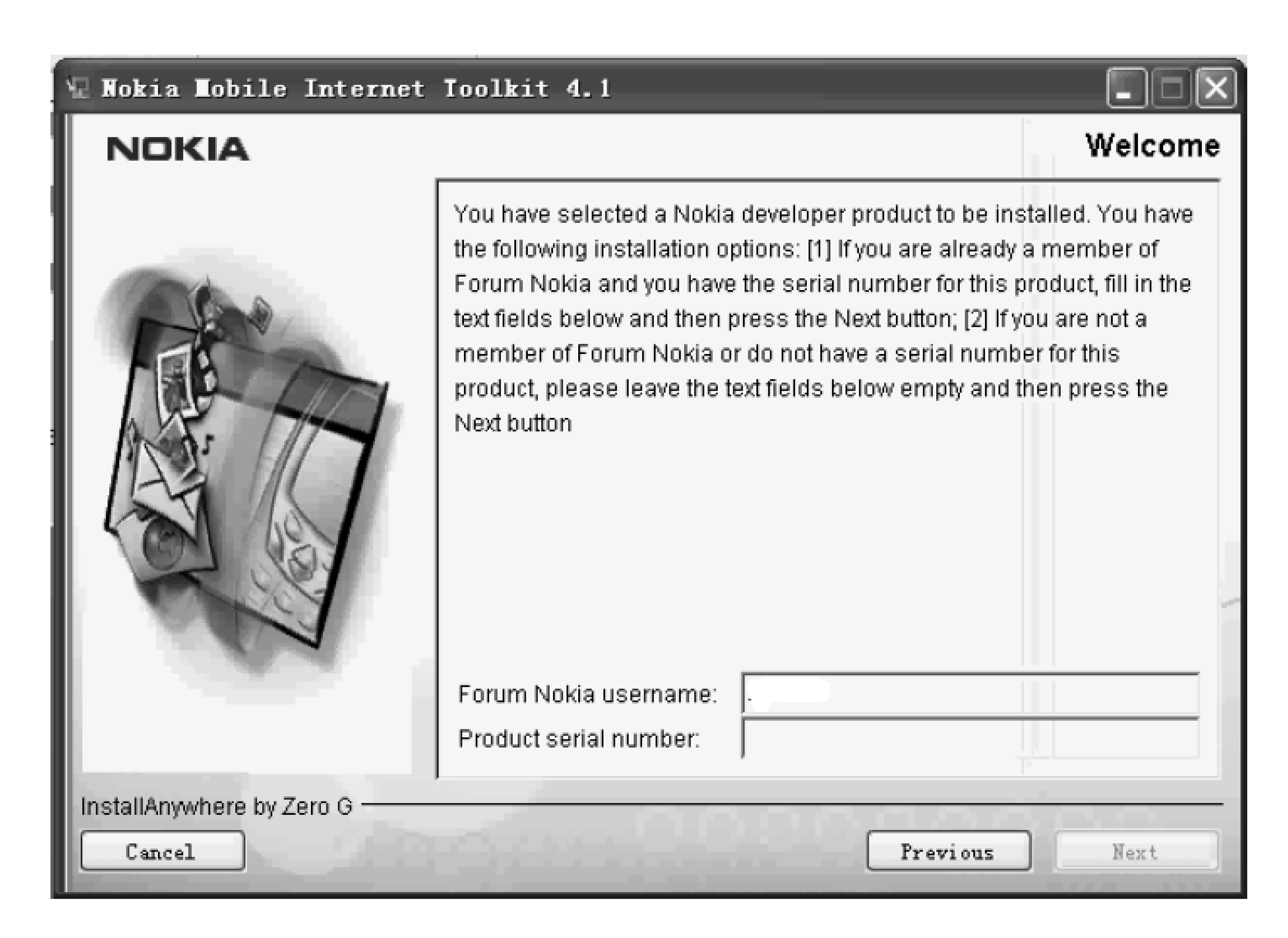


图 6-7　欢迎界面

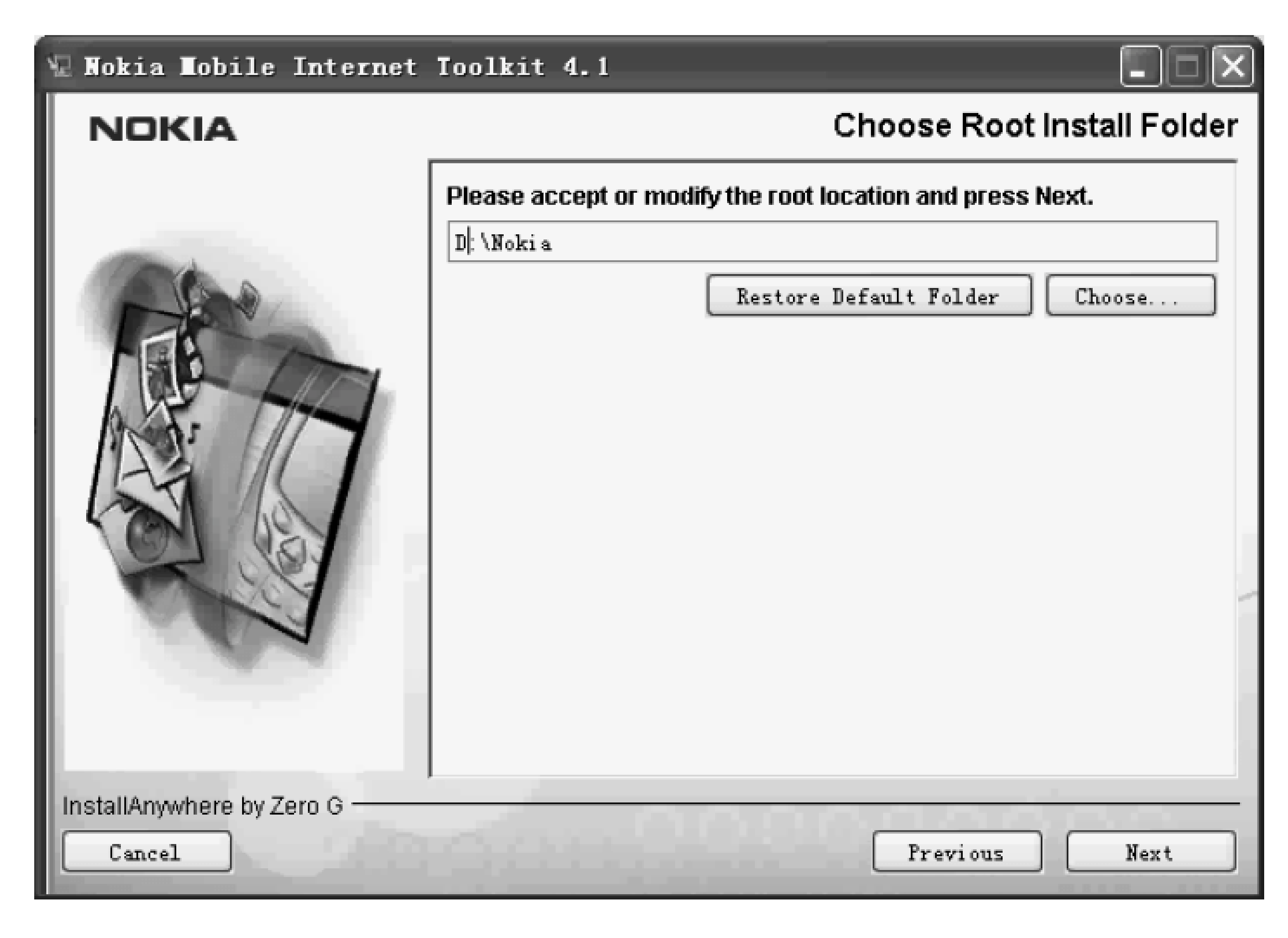


图 6-8　设置根目录

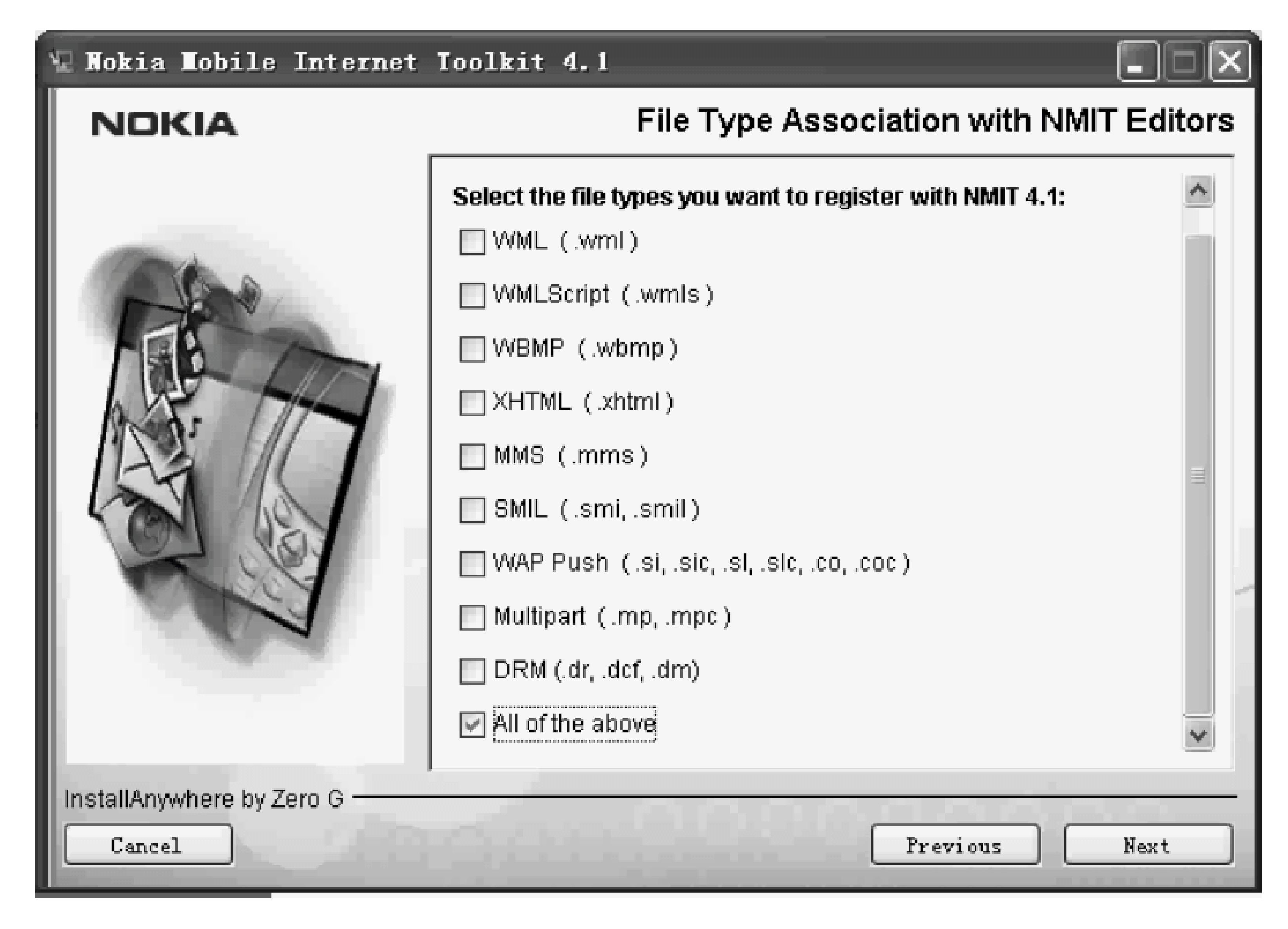


图 6-9　设置关联文件类型

⑦ 最后 Nokia Mobile Internet Toolkit 4.1 会安装并合并文件，直至结束安装。

(3) 安装 Patch for JRE 5.0 补丁文件，会出现一个启动画面，如图 6-10。当启动过程完毕，会进入到 Patch for JRE 5.0 的下一个安装步骤。然后按照与 Nokia Mobile Internet Toolkit 安装类似的步骤继续安装，直至安装结束。

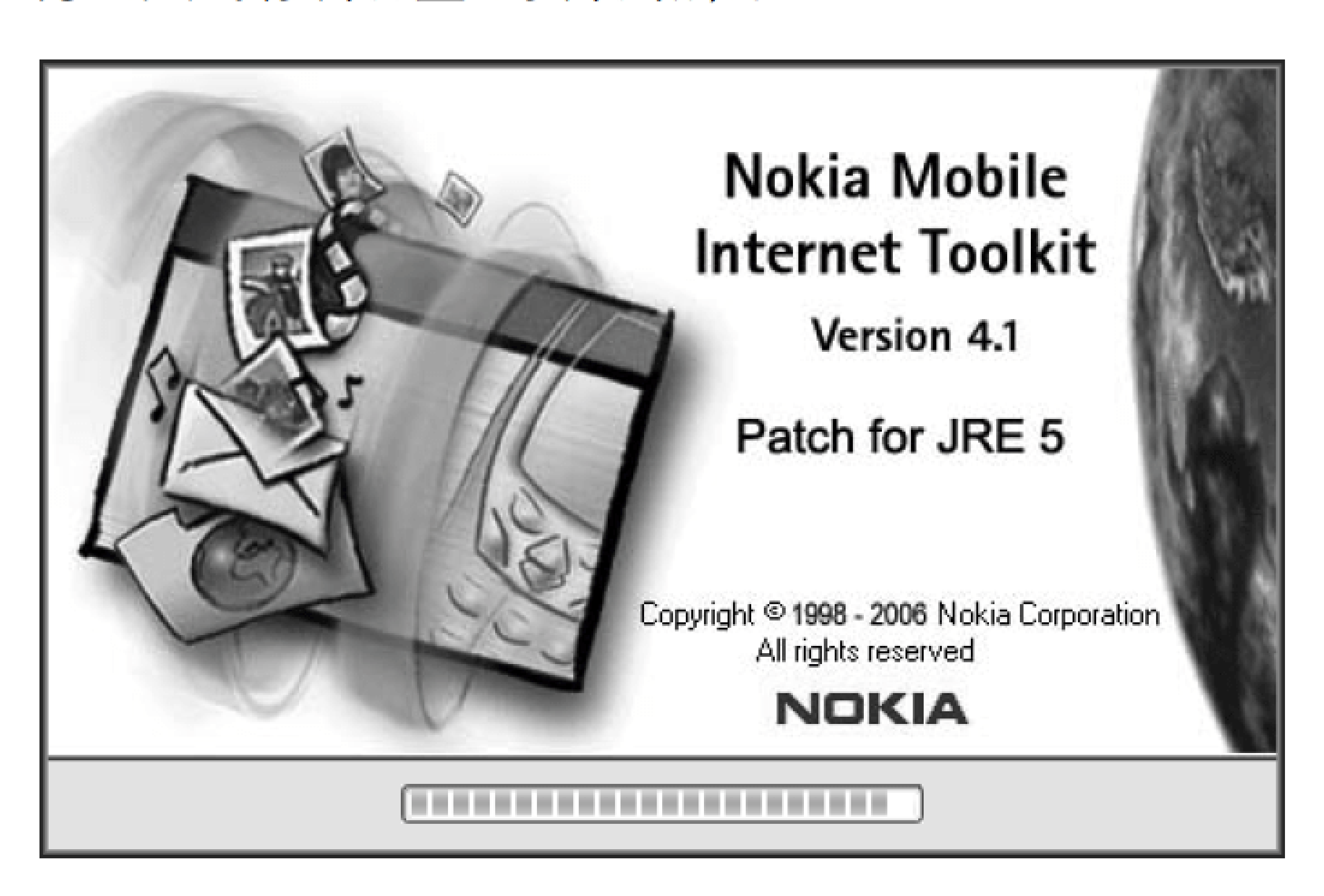


图 6-10　Patch for JRE 5.0 的启动

(4) 安装编辑器成功后，从 Windows 环境中选择“开始”|“程序”|Nokia Developer Tools|Nokia Mobile Internet Toolkit|NMIT 4.1，打开编辑器，如图 6-11 所示。接着，就可以在编辑器中用菜单栏的“文件”或左窗口中的导航功能实现新建、打开等与 WAP 网页制作相关的文件操作。

练习 2：安装和使用 WinWAP for Windows 4.1。

WinWAP for Windows 4.1 是芬兰 WinWAP 研发的浏览器软件。该软件可以在 Windows 环境下模拟手机浏览器的软件。该软件可支持 XHTML MP 1.2、XHTML 1.0、WML 2.0、HTML 4.01 和 CSS，是与 WAP 2.0 完全兼容的浏览器软件。该软件可以在

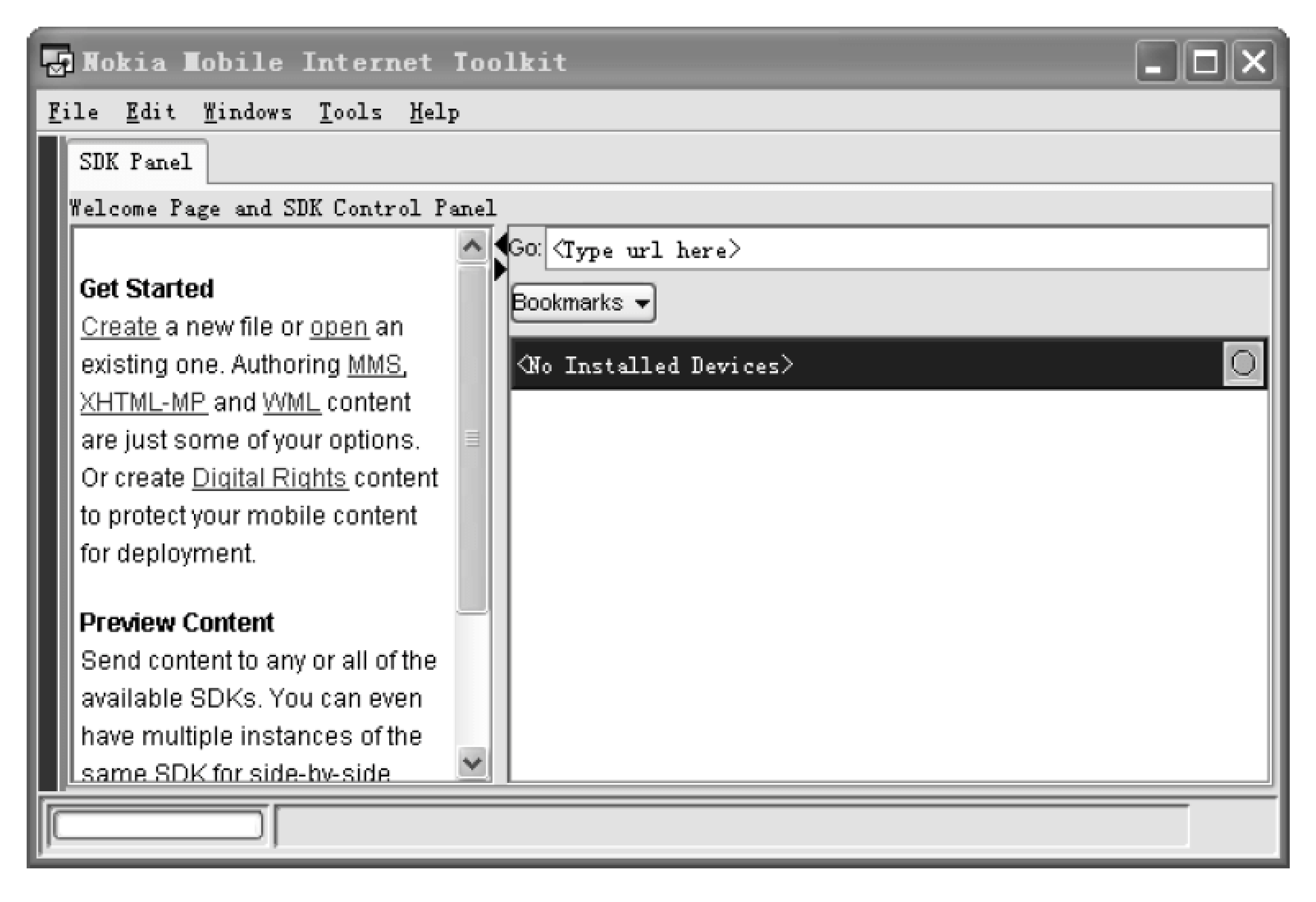


图 6-11　打开 Nokia Mobile Internet Toolkit 4.1

http://www.winwap.com/downloads 网站下载。

(1) 安装 WinWAP for Windows 4.1,双击下载后的 WinWAP-win32.exe 文件后,进入安装流程,过程如下。

① 首先出现一个选择安装语言的窗口,如图 6-12 所示。然后,用户根据自己的要求,在下拉列表中选择语言,单击 OK 按钮。

② 在随后出现的安装向导欢迎窗口(图 6-13)中单击 Next 按钮。进入安装目录设置窗口(图 6-14)。用户可单击 Browse 按钮选择安装目录,然后单击 Next 按钮,进行任务设置。



图 6-12　选择安装语言　　　图 6-13　安装向导欢迎窗口

③ 在"任务设置"窗口(图 6-15)中,有 3 个复选框,从上到下依次是"创建桌面快捷图标"、"创建一个快速启动图标"和"关联文件"。用户根据实际要求设置任务后,单击 Next 按钮,进入安装界面,如图 6-16 所示。安装完毕后,单击 Finish 按钮结束安装,如图 6-17 所示。



图 6-14　设置目录界面



图 6-15　任务设置

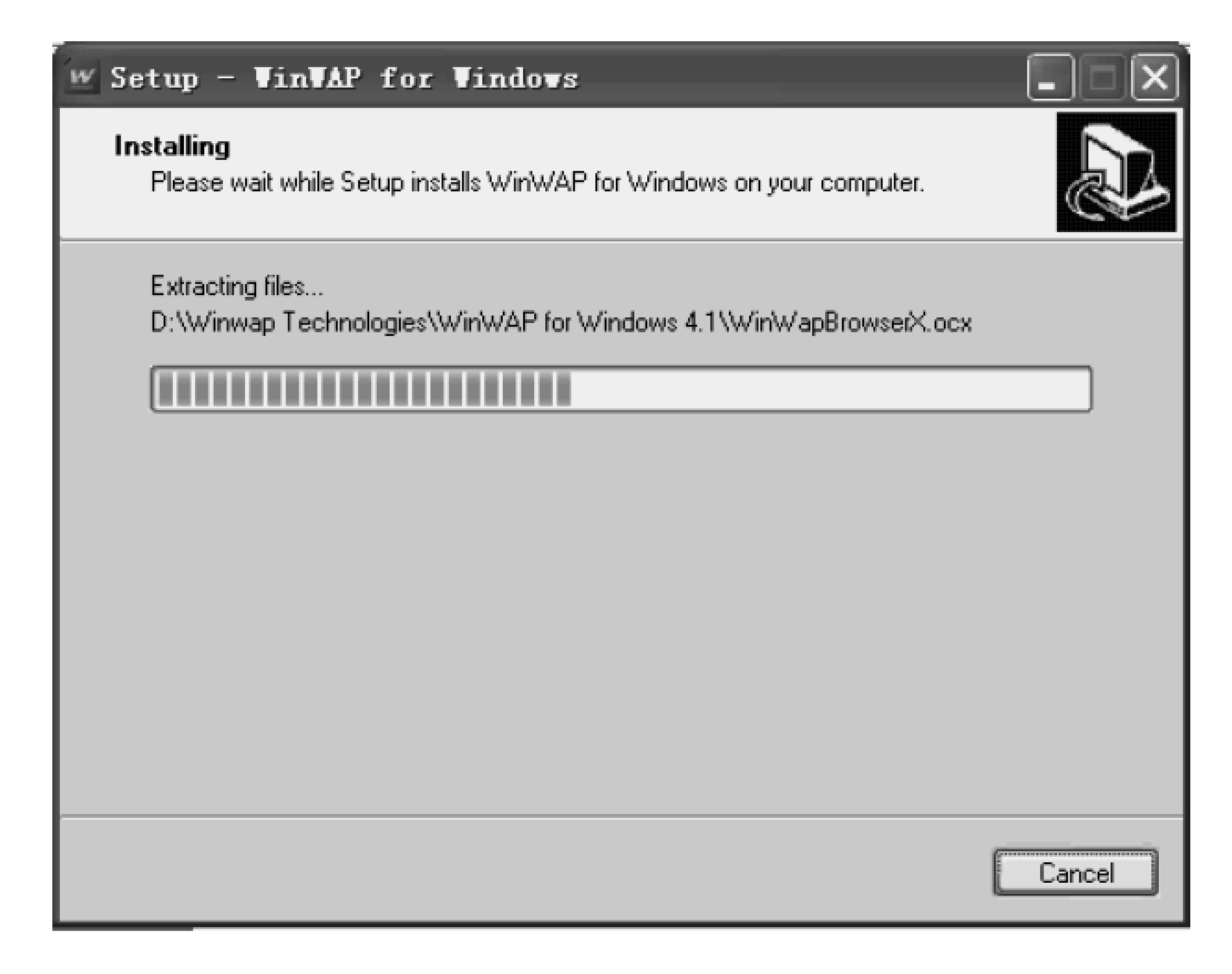


图 6-16　正在安装



图 6-17　完成安装

（2）使用 WinWAP for Windows 4.1。在 Windows 环境下，打开"开始"|"程序"|WinWAP for Windows|WinWAP for Windows 4.1 软件，出现图 6-18 的窗口。注意，在第一次使用该浏览器时需要设置语言。

如果使用的是"试用版"，需要获取"免费的试用证书"，否则无法浏览内容。具体做法是选择菜单栏中的"帮助"|"注册 WinWAP"项，进入"注册"窗口，如图 6-19 所示。单击 Get free evaluation（trial）license 单选按钮，再输入下载试用版 WinWAP 时使用的 E-mail 即可。

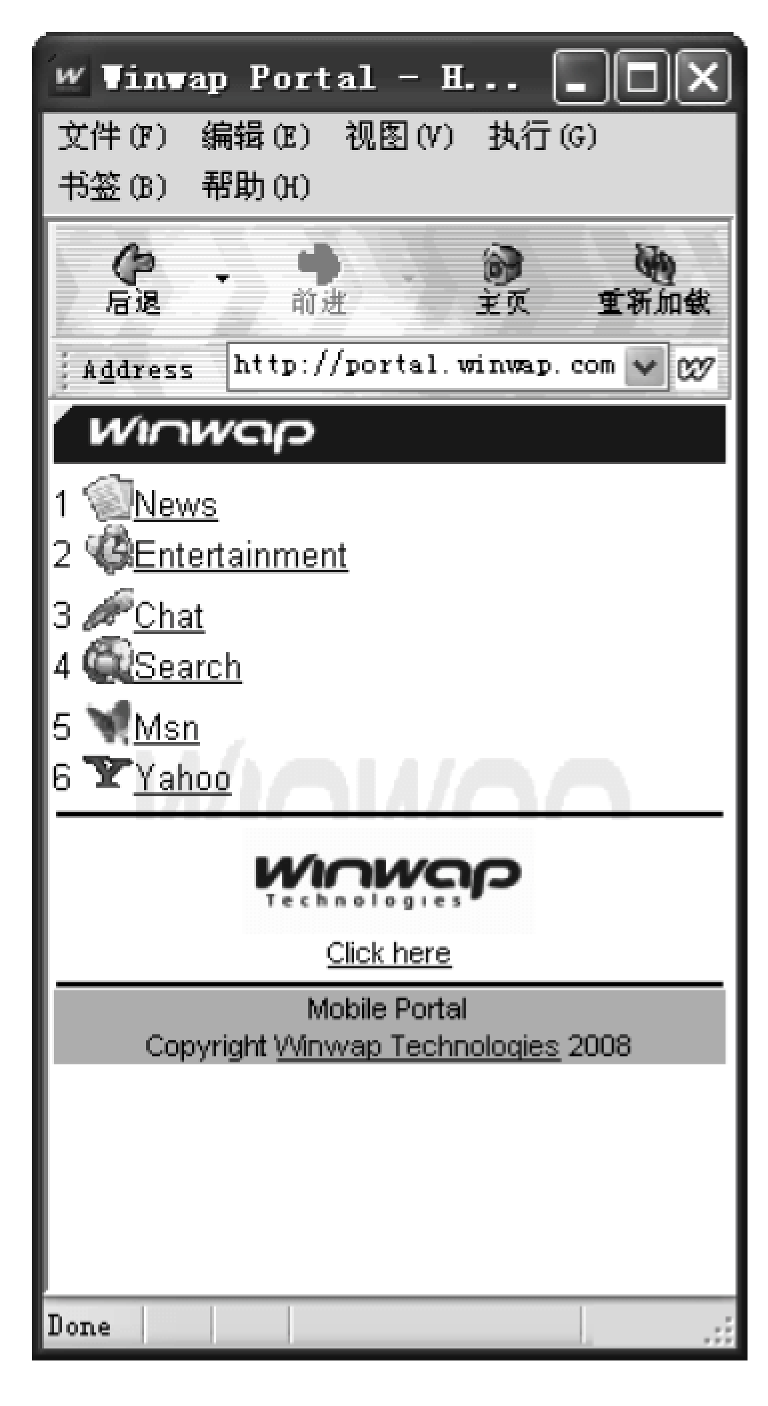


图 6-18　运行 WinWAP



图 6-19　注册窗口

如果具有注册证书，则选择 Get full license，并在 Registration Key 和 Your E-mail 中分别输入你购买的注册码和 E-mail。

浏览 WAP 网页是在 Address 栏中输入一个 WAP 网页的地址，如 http://wap.yahoo.com.cn，然后按 Enter 键，观察运行结果，并可以浏览该 WAP 网站的各个不同页面。

(3) 在 WinWAP 浏览器中打开 WML 1.x 的文件、XHTML MP 的文件以及 HTML 的文件，观察运行结果。说明哪些类型的文件可以正常显示。

练习 3：安装和使用 OpenWave SDK 6.2.2。

OpenWave SDK 6.2.2 是 OpenWave 公司开发的基于 Windows 平台的服务于无线应用的软件。OpenWave 内置一个手机模拟器 OpenWave Simulator，可以浏览无线 Web 应用。OpenWave SDK 6.2.2 可以从 http://www.developer/openwave 站点下载。如果需要运行 WMLScript 脚本的话，还需要下载 Phone Simulator 6.2.2（WAP 1.x 网络插件），支持 WAP 1.x 网络功能。

(1) 安装 OpenWave SDK 6.2.2。

① 双击下载的 Openwave_SDK_622.exe，进入安装向导窗口，运行状况如图 6-20。然后单击 Next 按钮。

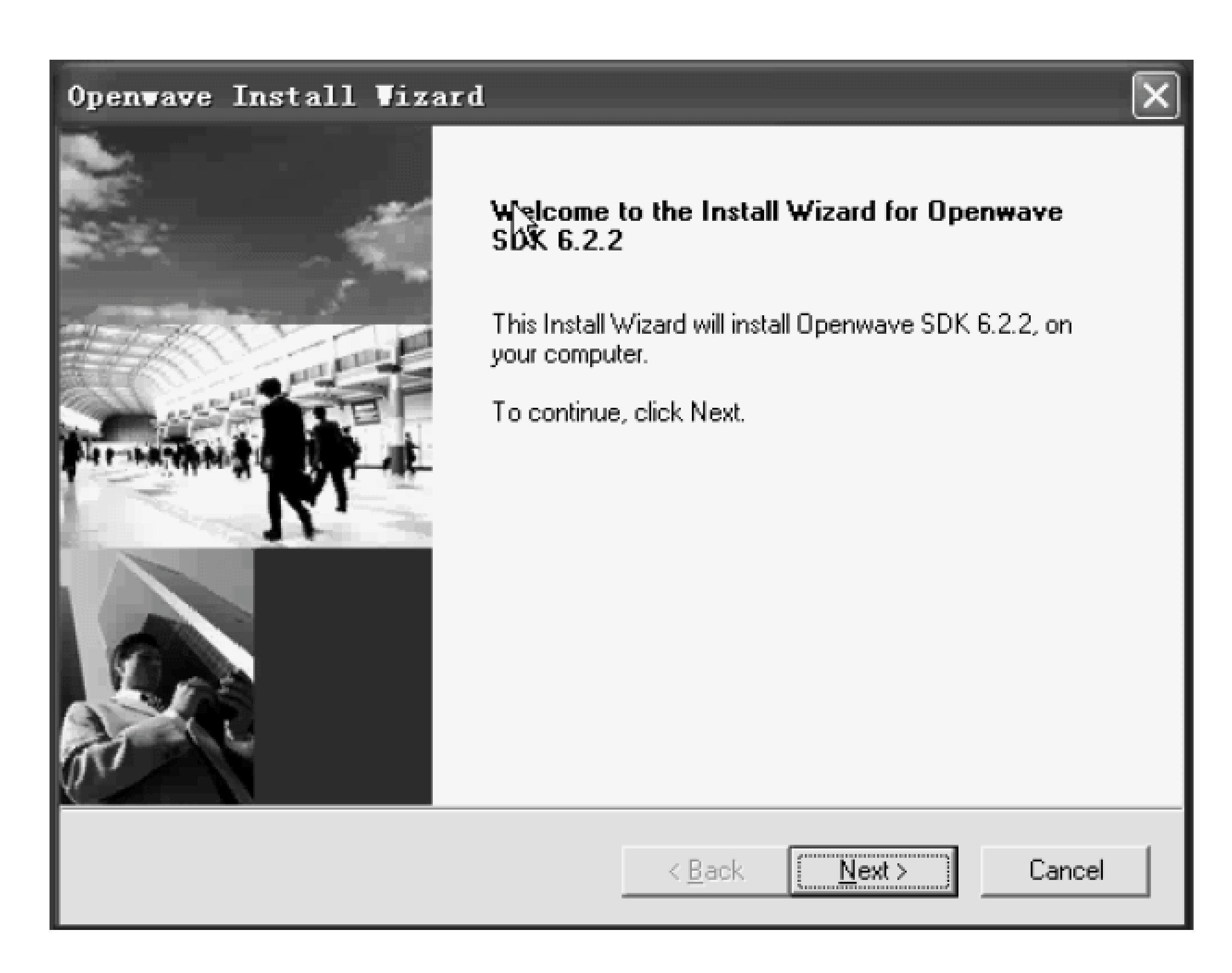


图 6-20　Openwave 安装向导

② 在出现的“安全协议”窗口，单击 Yes 按钮，如图 6-21 所示。

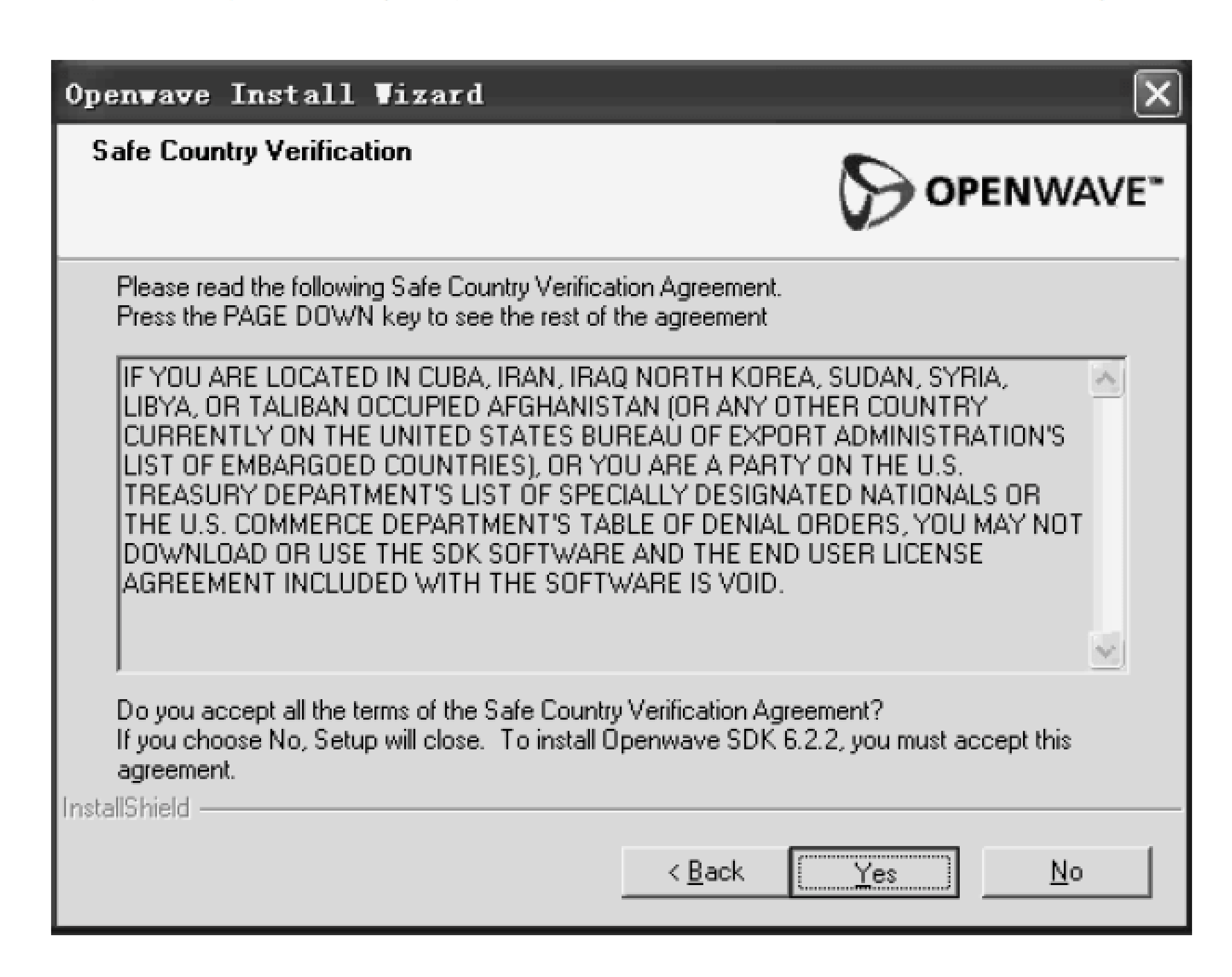


图 6-21　安全协议

③ 随后设置安装目录，单击 Browse 按钮，在目录列表中选定要安装的路径。设置完毕后，单击 Next 按钮，进入安装复制文件的过程。

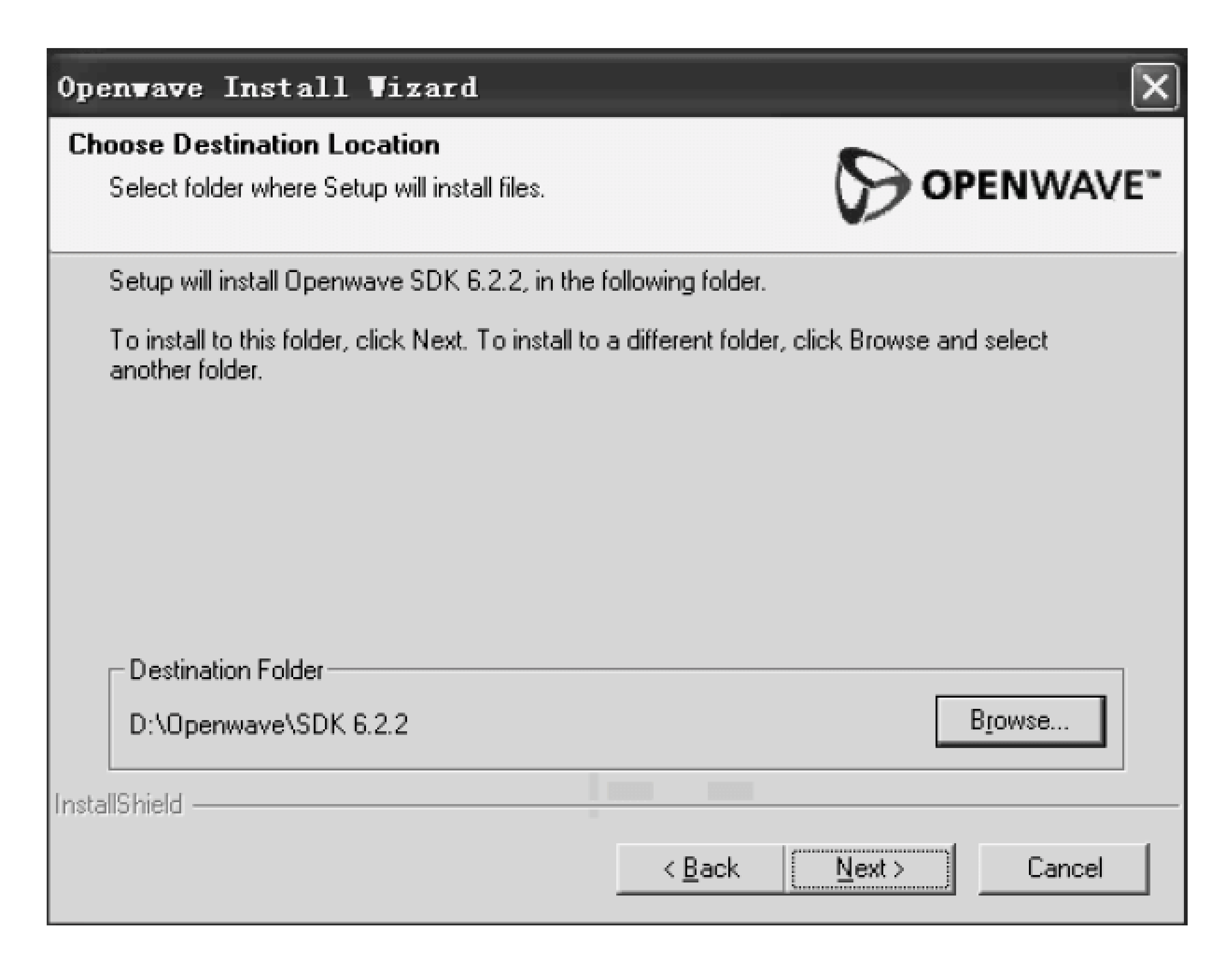


图 6-22　设置安装目录

④ 当所有的文件安装复制完成，会出现图 6-23 所示任务设置窗口。该窗口有两个复选框，分别表示“浏览发行说明”和“创建桌面快捷方式”。然后单击 Finish 按钮，结束安装。



图 6-23　安装完成

(2) 安装 Phone Simulator 6.2.2 WAP 1.x 网络插件。双击插件文件 Openwave_SDK_622wap.exe，操作过程类似安装“OpenWave SDK 6.2.2”。此处略。

注意： OpenWave 公司发布最新的浏览器是 OpenWave V7 Simulator，该浏览器模拟器与 WAP 2.0 规范保持一致，具有双协议功能，不需要另外安装支持 WAP 1.x 网络的插件。但是，OpenWave V7 Simulator 在本地不能支持 WMLScript，需要设置网关，特此说明。

(3) 使用 OpenWave SDK 6.2.2。在使用 OpenWave SDK 6.2.2 时，要注意无线应用的网络环境。如果无线应用是通过 HTTP 实现的，则打开“开始”|“程序”|OpenWave SDK 6.2.2|OpenWave SDK 6.2.2 HTTP；若使用的是 WAP 1.x 的 WAP 网关，则需要打开“开始”|“程序”|OpenWave SDK 6.2.2|OpenWave SDK 6.2.2 WAP。

① 打开 OpenWave SDK 6.2.2 HTTP。在 go 的地址栏中分别输入 http://wap.yahoo.com 和 http://wap.sina.com.cn，观察运行结果，并思考二者是否可以正常运行。并通过菜单栏的 Tools|Options，进入 SDK Configuration 配置窗口，在配置窗口中选择 Language，在出现的窗口中(图 6-24)设置语言和文字字体。

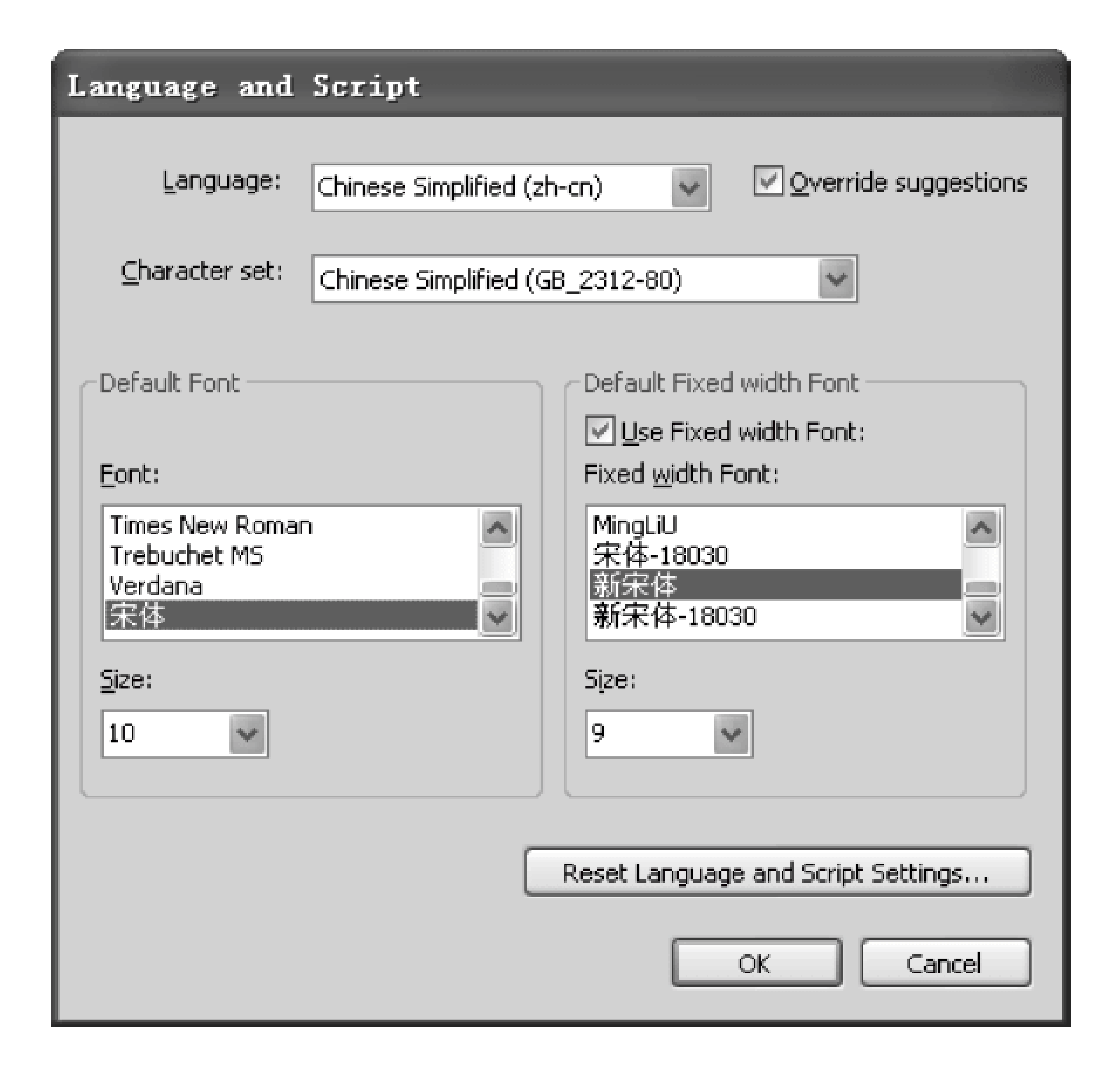


图 6-24 语言和脚本设置

② 打开 OpenWave SDK 6.2.2 WAP，同样在 go 地址栏中输入 http://wap.yahoo.com，观察并记录运行结果。

6.3 实验 6.2 WML 1.x 与 WMLScript 开发 WAP 应用

实验目的：

(1) 进一步掌握开发和测试 WAP 网页的软件工具。
(2) 熟练掌握 WML 1.x 的基本语法。
(3) 了解 WMLScript 的基本语法和常见的库函数的使用要求。
(4) 要求能结合 WML 1.x 和 WMLScript 解决具体的无线应用。
(5) 初步体验 WAP 网页的设计原则和基本要求。

实验内容：

(1) 完成一个 WAP 网页，使之实现矩形面积的计算。
(2) 为无线移动设备设计一个简易表单注册页面。

实验步骤：

练习 1：计算矩形的面积。

本次练习的目的是了解 WML 实现输入以及结合 WMLScript 实现计算和输出。具体要求：

（1）分别用 WinWap、OpenWave SDK 6.2.2 HTTP 和 OpenWave SDK 6.2.2 WAP 运行下列计算长方形面积的程序 area.wml，面积计算由 area.wmls 完成，它们的代码分别见程序清单 6-1 和程序清单 6-2，观察并分析运行结果，比较在不同浏览器下支持的功能有什么不同。

程序清单 6-1：

```
<?xml version="1.0" encoding="gb2312"?>
<!--area.wml-->
<wml>
    <card id="card1" title="面积" newcontext="true">
    <p>
        高度:<input format="N*M" name="height" title="高度:" value="0"/><br/>
        宽度:<input format="N*M" name="weight" title="宽度:" value="0"/>
        <br/>面积是：<b>$(result)</b>
        <do type="accept" name="cal" label="计算">
            <go href="area.wmls#calculate('$(height)', '$(weight)')"/>
        </do>
    </p>
    </card>
</wml>
```

程序清单 6-2：

```
/* area.wmls */
extern function calculate(height, weight) {
    var areaResult;
    areaResult=height*weight;
    WMLBrowser.setVar("result", areaResult);
    WMLBrowser.refresh();
}
```

（2）修改程序，使之用 OpenWave SDK 6.2.2 WAP 运行 area.wml 的结果如图 6-25 所示，实现矩形面积的计算。

注意： *在 OpenWave SDK 6.2.2 WAP 中运行的 WMLScript 脚本必须是已编译的，可以通过 Nokia Mobile Internet Toolkit 4.1 实现对文件的编译。*



图 6-25　矩形面积的计算

练习 2：设计和实现表单注册。

实现一个简易的表单注册，用户填写的注册信息有用户名、密码、E-mail、昵称和出生日期。要求对用户填写的注册信息进行验证，其中，用户名为必填项，不能为空；密码必选项，不能为空，并且密码的长度不能小于 8 个字符；E-mail 是必选项，填写形式要求合乎 xxx@com.cn 形式；出生日期是可选项，如填写必须合乎日期的格式的合法形式，形如“YYYYMMDD”（年月日）；昵称是可选项。要求如下：

(1) 下列程序 validateForm. wml(代码见程序清单 6-3)定义了注册表单页,所有信息格式的验证是通过 validateForm. wmlsc 来实现的。请将 代码 1 ～ 代码 3 处补充完整。

程序清单 6-3:

```
<?xml version="1.0" encoding="gb2312"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.3//EN"
    "http://www.wapforum.org/DTD/wml13.dtd">
<!--validateForm.wml-->
<wml>
<card id="card1" title="注册表单">
    <p>注意: * 是必填<br/>
     <b>$ (errorMsg)</b><br/>
     <!--出错信息-->
      * 用户名:<br/>
     <input name="username"/><br/>
      * 密码(不能小于 8 个字符)<br/>
     <input type= 代码 1  name="password"/><br/>
     <!--定义密码框-->
      * Email:<br/>
     <input name="email"/><br/>
     <!--定义 email 输入框-->
     昵称:<br/>
     <input name="name"/><br/>
     <!--定义昵称输入框-->
     出生日期 (YYYYMMDD):<br/>
     <input name="birthday" format= 代码 2  emptyok= 代码 3 /><br/>
     <!--定义文本框,只接收长度为 8 的数字字符-->
     <a href="validateForm.wmlsc#validate()">确定</a>
    </p>
</card>
</wml>
```

(2) validateForm. wmls 是验证输入数据的脚本,代码如程序清单 6-4 所示。阅读程序理解代码的含义,请将 代码段 1 ～ 代码段 4 处补充完整。

程序清单 6-4:

```
extern function validate(){
//验证格式是否正确
  var frmusername=String.trim(WMLBrowser.getVar("username"));
// 获取表单的用户名
  var frmpassword=String.trim(WMLBrowser.getVar("password"));
//获取表单的密码
  var frmemail=String.trim(WMLBrowser.getVar("email"));
//获取表单的 email
```

```
  var frmname=String.trim(WMLBrowser.getVar("name"));
//获取表单的 name
  var frmbirthday=String.trim(WMLBrowser.getVar("birthday"));
//获取表单的出生日期
  if (frmusername==""){
//判断用户是否为空
    WMLBrowser.setVar("errorMsg", "用户名不能为空");
    WMLBrowser.refresh();
    return;
  }

  if (frmpassword==""){
//判断密码是否为空
    WMLBrowser.setVar("errorMsg", "密码不能为空");
    WMLBrowser.refresh();
    return;
  }

  if (frmemail==""){
//判断 email 是否为空
    WMLBrowser.setVar("errorMsg", "Email 不能为空");
    WMLBrowser.refresh();
    return;
  }
```

代码段

```
/** 补充代码,处理判断密码是否长度小于 8,
* 如果长度小于 8 个字符,返回 validateForm.wml 页面并传回"密码长
* 度不能小于 8 个字符."信息
**/

  if (!isEmailValid(frmemail)){
//判断 email 格式是否正确
    WMLBrowser.setVar("errorMsg", "Email 的格式不合法");
    WMLBrowser.refresh();
    return;
  }

  if (""!=frmbirthday && !is DateValid(frmbirthday)){
//判断出生日期为空或是否格式正确
    WMLBrowser.setVar("errorMsg", "日期格式不正确");
    WMLBrowser.refresh();
    return;
  }
  submit_form(frmusername, frmpassword, frmemail, frmname, frmbirthday);
```

```
//输入信息格式正确,将数据发送下一个页面
}

function is DateValid(birthday) {
//判断日期格式是否正确
代码段 2
/**
* 补充代码,实现日期格式是否与 YYYYMMDD 格式一致,
* 是,则返回 true,
* 否,则返回 false;
**/
}

function is EmailValid(emailAddr){
//验证 email 格式
代码段 3
/**
* 补充代码,实现 email 格式合法性的验证,
* 合法,返回 true,
* 否则,返回 false;
**/
}

function submit_form(frmusername, frmpassword, frmemail, frmname, frmbirthday){
  //发送表单数据至 success.wml
  WMLBrowser.setVar("errorMsg", "");
  代码段 4
/**
* 补充代码,将用户名、密码、E-mail、昵称、出生日期
* 的信息发送到下一个页面中。
**/
WMLBrowser.go("success.wml");
}
```

6.4 实验 6.3 XHTML MP 和 WCSS 开发 WAP 应用

实验目的:

(1) 了解和掌握 XHTML MP 语法和常使用的标记。
(2) 比较 XHTML MP 和 WML 1.x 以及 WML 2.0 的不同之处。
(3) 了解和使用 WCSS 设计 WAP 网页的样式,了解设计样式的基本规则。
(4) 结合 XHTML MP 和 WCSS 开发无线应用。

实验内容：

（1）为某门户 WAP 网站设计菜单，要求实现各种类型菜单的制作。

（2）为提供的心理测试素材设计，并实现一个简单心理测试的文本游戏。

实验步骤：

练习 1：菜单页面的设计。

本次练习是设计一个二级菜单。第一级菜单项有“新闻”、“游戏”、“股票”、“无线论坛”、“生活”。“新闻”菜单项的第二级菜单有“时事新闻”、“娱乐新闻”、“财经新闻”、“体育新闻”、“国际新闻”；“游戏”菜单项导向的第二级菜单有“RGP 游戏”、“动作游戏”、“益智游戏”、“棋牌游戏”；其他菜单项目为空。要求如下：

（1）下列 main. xhtml（代码见程序清单 6-5）程序定义了用文本链接实现第一级菜单，要求为 main. xhtml 定义一个样式表 style. css，使得显示的文本颜色为蓝色，并且对应的每一个链接都有快捷按钮，按数字“1”快速导航到 newmenu. xhtml，按数字“2”快速导航到 stockmenu. xhtml 网页，按数字“3”快速导航到 gamemenu. xhtml，按数字“4”快速导航到 forum. xhtml，按数字“5”快速导航到 life. xhtml。

程序清单 6-5：

```
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD XHTML Mobile 1.0//EN"
"http://www.wapforum.org/DTD/xhtml-mobile10.dtd">

<!--main.xhtml-->
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>文本菜单</title>
<link rel="stylesheet" href="style.css"/>
</head>

<body>
<div>
    <h4>欢迎 WAP 世界</h4>

    <table width="80%">
    <tr>
        <td><a    class="control_1" href="newsmenu.xhtml">新闻</a></td>
    </tr>
    <tr>
        <td><a    class="control_2" href="stockmenu.xhtml">股票</a></td>
    </tr>
    <tr>
        <td><a class="control_3" href="gamemenu.xhtml">游戏</a></td>
    </tr>
```

```
    <tr>
        <td><a class="control_4" href="forum.xhtml">无线论坛</a></td>
        </tr>
    <tr>
        <td><a class="control_5" href="life.xhtml">生活</a></td>
     </tr>
</table>

</div>
</body>
</html>
```

考虑自行设计图标，为 main. xhtml 网页的每一个菜单项增加一个图标。菜单项仍是分行显示。修改 style. css，要求为所有的图标设置大小：宽和高均为 8px。

注意：WCSS 的软键盘-wap-accesskey 属性只对 4 个元素 a、input、label 和 textarea 发生作用，对于其他元素是无效的。

（2）编写 newmenu. xhtml 网页，该网页用选择列表定义“新闻”的二级菜单实现，运行结果类似图 6-26。

（3）编写 gamemenu. xhtml，用有序列表来定义“游戏”的二级菜单，运行结果类似图 6-27。




图 6-26　新闻的选择列表二级菜单



图 6-27　游戏的有序列表二级菜单

（4）将上述类型的菜单用 WML 2.0 重新编写，比较 WML 2.0 与 XHTML Mobile 之间的异同。

思考：比较上述 4 种类型的菜单的异同，写出 4 种不同菜单的特点。并分析在实际无线应用中菜单设计的决定因素是什么。

练习 2：设计并实现一个趣味心理游戏。

设计一个趣味心理游戏，游戏由 10 个题目构成，通过回答不同答案导向不同的页面，最终得出测试的结果。提供素材的"心理测试题.txt"。要求游戏设计合理，并为游戏设计合理的文字、布局、图片等样式，界面设计可用性好。

操作步骤略。

第 7 章　JSP 开发的 Java 语言基础

在 JSP 开发中,几乎所有的服务器端动态功能都需要用 Java 语言来实现,JSP 开发中的 JavaBean、Servlet、Struts 等功能的实现基础也是 Java 语言。所以能否熟练地编写 Java 程序关系到能否高效地进行 JSP 开发。本章就基本的 Java 语言设计相关实验,力求读者通过这些实验检验自己的 Java 编程能力,为以后的 Web 程序设计打好基础。

7.1　预 备 知 识

7.1.1　Java 简介

Java 语言是由 Sun Microsystems 公司开发的一种面向对象程序设计语言,Java 语言吸收了 Smalltalk 语言和 C++ 语言的优点,并增加了其他特性,如支持并发程序设计、网络通信和多媒体数据控制等。主要特性如下:

(1) Java 语言是简单的。

(2) Java 语言是面对对象的。

(3) Java 语言是分布的。

(4) Java 语言是健壮的。

(5) Java 语言是安全的。

(6) Java 语言是体系结构中立的。

(7) Java 语言是可移植的。

(8) Java 语言是解释型的。

(9) Java 语言是高性能的。

(10) Java 语言是多线程的。

(11) Java 语言是动态的。

7.1.2　Java 的基本语法

1. 数据类型

Java 支持多种数据类型,它们可以用来声明变量、创建数组以及其他更复杂的数据结构。Java 的数据类型如图 7.1 所示。

2. 数组

在 Java 程序设计中,可以用数组来表示一些相关的数据。数组是多个数据项的有序集合,其中的每个数据项称为数组的元素。在 Java 程序中,数组具有下列特点。

(1) 同一数组中的所有元素均属于相同的数据类型,该数据类型称为数组的基本类型。

(2) 数组一经创建,其元素个数就保持不变,这个长度称为数组的长度。

(3) 数组中的每一个元素均能借助于下标(index)来访问。

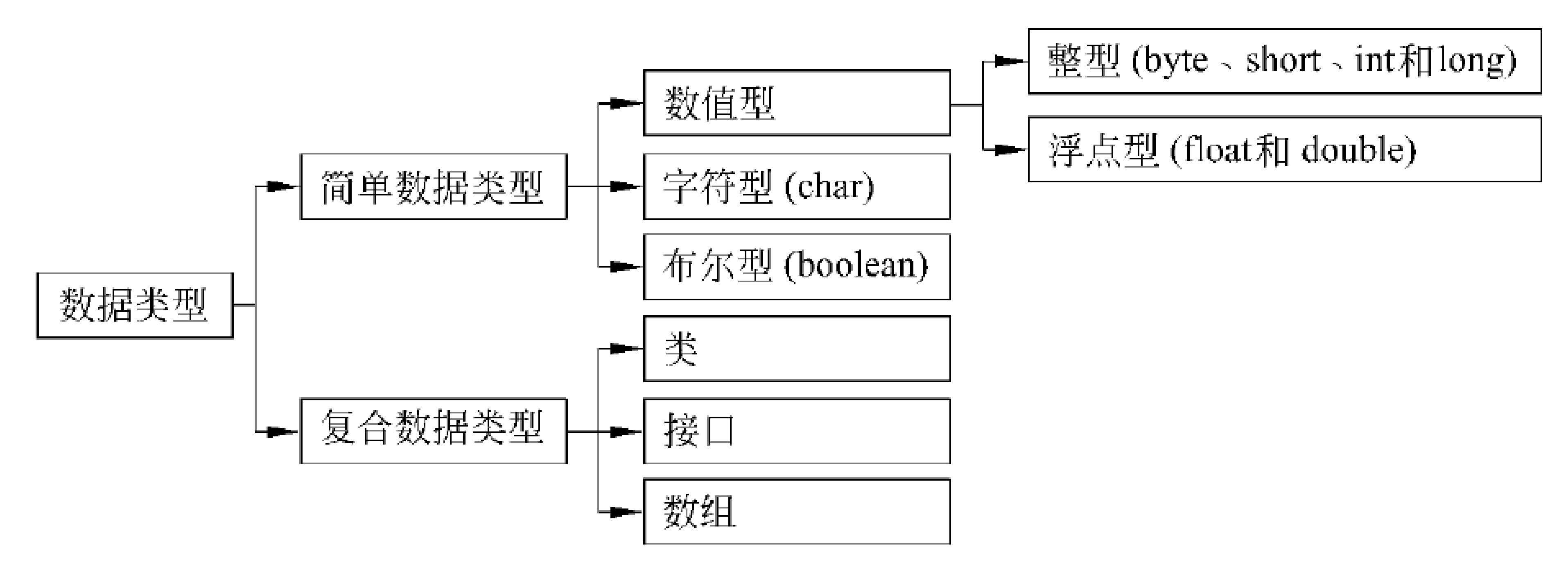


图 7-1　Java 数据类型

(4) 数组元素的类型既可以是简单数据类型(如 int 和 float 等),也可以是复合类型(如 String、Object,甚至数组类型)。元素类型如果属于简单数据类型则每个元素都为一个值,如果属于复合类型则每个元素是一个对象。

在 Java 语言中的数组按照以下方式声明、创建和引用。

(1) 声明形式如下:

```
type[][]… arrayName;
```

或者

```
type arrayName[][]…;
```

(2) 创建。声明数组时,并没有为其中的元素分配存储空间,而要等到创建数组的时候才真正为它们分配存储空间。在 Java 中,数组的创建必须使用 new 操作符:

```
arrayName=new elementType[ARRAY_SIZE1] [ARRAY_SIZE2]…;
```

(3) 引用。利用数组的下标来使用数组元素:

```
arrayName [index1] [index2]…
```

3. 常用运算

Java 语言中,运算符可以划分为四大类:算术运算符、关系运算符、位运算符以及逻辑运算符。

4. 控制语句

Java 语言中,依靠 if 语句、switch 语句、for 语句、while 和 do…while 语句来控制程序结构,这些语句的语法和 C、C++ 语言基本一致。

7.1.3　Java 面向对象编程基础

Java 是一种纯粹的面向对象程序设计语言,在 Java 中,所解决问题中的所有的东西都是对象,所以 Java 编程实际上就是对类和对象的编程。

1. 类和对象

Java 语言中,定义一个类的语法形式如下:

```
class 类名{
    成员属性;
```

```
    成员方法;
}
```

Java 语言中,需要为类创建对应的实例对象才能使用,创建对象的语法形式如下:

```
类名 对象名=new 类名([参数列表]);
```

2. 继承性

在 Java 中,通过继承,子类拥有父类的所有成员,包括成员变量和成员方法,一个类 A 继承另一个已存在的类 B 的语法形式如下:

```
class A extends B {
    成员列表;
}
```

注意: Java 语言只支持单继承,不允许多继承。

3. 包

Java 语言引入包(package)的机制使得 Java 更易于层次化地组织大型的 Java 程序。

定义包的语句格式:

```
package 程序包名;
```

引用包的语句格式:

```
import 包名;
```

4. 接口

Java 是不支持多继承的,但是 Java 可以通过实现接口(interface)来实现多继承。接口是一种特殊的抽象类,这种抽象类中只包含常量和方法的声明,而没有变量和方法的实现。

接口的声明语法形式:

```
interface 接口名{
    成员列表;
}
```

接口的实现语法形式:

```
class 类名 implements 接口名
```

5. 多态性

多态性就是程序中同一个符号在不同的情况下具有不同的意义。

多态性分为编译时多态性和运行时多态性,编译时多态叫静态绑定,由简单的方法重载实现。运行时多态也叫动态绑定。多态性的原理是在程序运行期间判断所引用对象的实际类型,根据其实际类型调用相应的方法。多态的存在必须具备 3 个条件。

(1) 有继承。程序中存在子类对父类的继承(包括实现接口)。

(2) 有覆盖。子类必须对父类中的必要方法进行重写,以实现自己需要的功能。

(3) 父类引用指向子类对象。程序调用的必须是父类,而实际上编译器却调用子类的对象。

7.1.4 Java 的异常处理

1. 异常与异常类

异常就是程序运行过程中遇到的错误，使程序运行中止，或者即使程序能够继续运行，但得出错误的结果甚至导致严重的后果。引入异常处理机制后，异常处理就可以使用统一的语句和接口，使错误处理更方便、更安全。

Java 语言中通过定义异常类来表示各种错误。

Exception：所有异常类的父类，其子类对应各种各样可能出现的异常事件，一般需要用户显式地声明或捕获。

其他异常类中比较常见的有 IOException、InterruptedException、ClassNotFoundException、DataFormatException、SQLException 等。

2. 异常的抛出

Java 程序的执行过程如出现异常事件，可以生成一个异常类对象，该异常对象封装了异常事件的信息并将被提交给 Java 运行时系统，这个过程称为抛出(throw)异常。

异常抛出的语法形式：

```
throw 异常类对象;
```

3. 捕获异常

当 Java 运行时系统接收到异常类对象时，会寻找能处理这一异常的代码并把当前异常对象交其处理，这一过程称为捕获(catch)异常。

异常捕获的语法结构如下(try catch finally)：

```
try {
    可能出现特定异常的代码;
} catch() {
    处理异常的语句 1;
} catch() {
    处理异常的语句 2;
}
    ⋮
    finally {
      最终执行的语句;
}
```

7.1.5 Java 的多线程

1. 多线程的定义

Java 语言支持多线程的程序，多线程编程是指将程序任务分成几个并行的子任务，由这些子任务并发执行，一起协作完成程序的功能。多线程的执行是并发的，即在逻辑上是“同时”的，而不管是否是物理上的“同时”。

在 Java 语言中，线程的创建可以用两种方法，第一种是定义 Thread 的子类，另一种是实现 Runnable 接口。

(1) 继承 Thread 类并重定义 run()方法。定制一个线程在运行时执行的代码,可以通过定义一个 Thread 类的子类,并重定义从 Thread 中继承过来的 run()方法,使之执行指定的代码。run()方法是 Thread 中定义的一个空方法,定义格式如下:

```
class MyThread extends Thread {
    public void run() {…}                    //重写 run 方法
}
```

当启动一个新线程时,run()方法中的第一条语句就是新线程将执行的第一条语句。线程启动的代码如下:

```
MyThread myThread=new MyThread(…);
Thread t=new Thread(myThread);
t.start();
```

(2) 定义线程类实现 Runnable 接口。Runnable 接口中只有一个 run 方法,任何类只要实现了 Runnable 接口,系统就会自动识别出它为一个线程类。定义的格式如下:

```
class MyThread implements Runnable {
    public void run() {…}                    //覆盖 run 方法
}
```

启动新线程的方法与上面相同。

2. 线程的生命周期

从创建一个新线程到这个线程消亡(或终止)的时间段称为线程的生命周期。在整个生命周期中,由于受到外界或线程自身需要的影响,线程表现出 5 种不同的状态。

(1) 创建状态。

(2) 可运行状态。

(3) 运行中状态。

(4) 阻塞状态。

(5) 死亡状态。

3. 线程的优先级

线程的优先级用数字表示,范围为 1~10,一个线程的默认优先级是 5。线程类中用 3 个常量记录 3 种不同的优先级:Thread.MIN_PRIORITY、Thread.MAX_PRIORITY 和 Thread.NORM_PRIORITY,它们分别对应优先级 1、10 和 5。

7.2 实验 7.1 Java 的异常处理

实验目的:

(1) 掌握 Java 中异常处理机制。

(2) 熟悉 try…catch…finally 语句结构和捕捉处理异常的方法。

(3) 了解有异常处理和没有异常处理的区别。

(4) 多重 catch 语句的使用。

实验内容：

本次实验训练读者掌握Java的异常和异常处理方法，实验完成了一个基于GUI的除法运算，要求读者分析并处理其中的异常情况。本实验由3个练习构成。

（1）分析程序，分析除法运算代码中的异常情况。

（2）为了增加程序的健壮性，利用try…catch处理异常。

（3）设计人性化的异常处理代码，提示用户程序中出现异常的原因。

实验步骤：

练习1：分析程序，分析除法运算代码中的异常情况。

如图7-2，运行并测试程序lab7_1_1.java，测试20/10、20/0、20/abc看这个程序是否会产生异常；会产生哪些异常；程序有没有对异常进行处理；一旦出现异常将会对程序的执行造成什么样的影响？

图7-2　Java除法计算

程序清单7-1：

```
//lab7_1_1.java
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class Lab7_1_1 extends JFrame {
    private JTextField jtfNum1, jtfNum2, jtfResult;
    private JButton jbtDiv=new JButton("Divide");

    public Lab7_1_1() {
        JPanel p1=new JPanel();
        p1.setLayout(new FlowLayout());
        p1.add(new JLabel("被除数"));
        p1.add(jtfNum1=new JTextField(3));
        p1.add(new JLabel("除数"));
        p1.add(jtfNum2=new JTextField(3));
        p1.add(new JLabel("答案"));
        p1.add(jtfResult=new JTextField(4));
        jtfResult.setEditable(false);
        jtfResult.setHorizontalAlignment(SwingConstants.RIGHT);
        this.getContentPane().add(p1, BorderLayout.CENTER);
        this.getContentPane().add(jbtDiv, BorderLayout.SOUTH);
        jbtDiv.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                handleDivide(e);
            }
        });
    }
```

```java
    private void handleDivide(ActionEvent e) {
        if (e.getSource() ==jbtDiv) {
            int num1=Integer.parseInt(jtfNum1.getText().trim());
            int num2=Integer.parseInt(jtfNum2.getText().trim());
            int result=num1 / num2;
            jtfResult.setText(String.valueOf(result));
        }
    }

    public static void main(String[] args) {
        Lab7_1_1 exe=new Lab7_1_1();
        exe.pack();
        exe.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        exe.setVisible(true);
    }
}
```

练习 2：利用 try…catch 处理异常。

为了能够处理 20/0 或者 20/abc 时出现的异常，在练习 1 中 lab7_1_1.java 代码加入处理异常的 try…catch 块，修改好的代码程序见清单 7-2，分析并运行程序 lab7_1_2.java，回答执行 20/0 和 20/abc 时程序是否还会出现异常，一旦出现异常将会对程序的执行造成什么样的影响？

程序清单 7-2：

```java
//Lab7_1_2.java
import java.awt.* ;
import java.awt.event.* ;
import javax.swing.* ;

public class Lab7_1_2 extends JFrame {
    private JTextField jtfNum1, jtfNum2, jtfResult;
    private JButton jbtDiv=new JButton("Divide");

    public Lab7_1_2() {
        JPanel p1=new JPanel();
        p1.setLayout(new FlowLayout());
        p1.add(new JLabel("被除数"));
        p1.add(jtfNum1=new JTextField(3));
        p1.add(new JLabel("除数"));
        p1.add(jtfNum2=new JTextField(3));
        p1.add(new JLabel("答案"));
        p1.add(jtfResult=new JTextField(4));
        jtfResult.setEditable(false);
        jtfResult.setHorizontalAlignment(SwingConstants.RIGHT);
```

```java
        this.getContentPane().add(p1, BorderLayout.CENTER);
        this.getContentPane().add(jbtDiv, BorderLayout.SOUTH);

        jbtDiv.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                handleDivide(e);
            }
        });
    }

    private void handleDivide(ActionEvent e) {
        if (e.getSource() ==jbtDiv) {
            try {
                int num1=Integer.parseInt(jtfNum1.getText().trim());
                int num2=Integer.parseInt(jtfNum2.getText().trim());
                int result=num1 / num2;
                jtfResult.setText(String.valueOf(result));
            } catch (Exception ex) {
            }
        }
    }
    public static void main(String[] args) {
        Lab7_1_2 exe=new Lab7_1_2();
        exe.pack();
        exe.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        exe.setVisible(true);
    }
}
```

练习 3：设计人性化的异常处理代码，提示用户程序中出现异常的原因。

练习 2 中的程序仅仅只是完成了异常的捕捉，没有处理异常，这样可以防止程序中出现异常影响程序的正常运行。请判断并写出练习 2 中出现的异常类型。

修改练习 2 中的程序 Lab7_1_2.java，增加可能会出现异常的处理代码，即填充下列的程序清单 7-3 中 代码 1 ～ 代码 4 ，使得运行结果如图 7-3 所示。在实验 2 中加入程序清单 7-3 所示代码。

程序清单 7-3：

```java
⋮
private void handleDivide(ActionEvent e) {
    if (e.getSource()==jbtDiv) {
        try {
            int num1=Integer.parseInt(jtfNum1.getText().trim());
            int num2=Integer.parseInt(jtfNum2.getText().trim());
```

```
                int result=num1/num2;
                jtfResult.setText(String.valueOf(result));
            } catch (代码 1) {
                代码 2;
            } catch (代码 3) {
                代码 4;
            }
        }
    }
    ⋮
```

保存新的代码为 Lab7_1_3.java,运行并测试程序,可以观察到如图 7-3 所示的执行效果。



(a) 正确执行的效果





(b) 当除数是 0时报错的情况





(c) 数字格式错误报错的情况

图 7-3　Java 异常处理

7.3　实验 7.2　Java 的多线程处理

实验目的:

(1) 掌握 Java 中多线程程序的原理。

(2) 熟悉利用 Thread 或 Runnable 实现多线程程序的方法。

(3) 了解线程的生命周期。

实验内容：

本次实验训练读者掌握 Java 的多线程编程方式，实验由两个练习构成。

（1）编写程序，生成一个蓝色的反弹球，每当碰到边框的时候，被弹起并沿反方向运动。

（2）修改上面的程序，允许用户增加或减少新的球。最多提供 20 个球，为每个球随机选择颜色。

实验步骤：

练习 1：编写程序，生成一个蓝色的反弹球，每当碰到边框的时候，被弹起并沿反方向运动。

（1）启动 Java 开发环境，新建一个文件 Lab7_2_1.java，将程序清单 7-4 代码输入。

程序清单 7-4：

```
//Lab7_2_1.java
import java.awt.* ;
import javax.swing.* ;
import javax.swing.border.LineBorder;
public class Lab7_2_1 extends JFrame {
    public Lab7_2_1(String title) {
        super(title);
        this.getContentPane().add(new BallJumpPan());
        this.pack();
    }
    public static void main(String args[]) {
        Lab7_2_1 exe=new Lab7_2_1("多线程实验");
        exe.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        exe.setVisible(true);
    }
    private class Ball {
        int xb, yb;
        int vx, vy;
        boolean state=true;
        final int W=5, H=5;
        Color color;
        public Ball(int x, int y) {
            xb=x;
            yb=y;
            vx=-(int) (Math.random() * 10+5);
            vy=-(int) (Math.random() * 10+5);
            color=Color.BLUE;
        }
    }
    private class BallJumpPan extends JPanel implements Runnable {
```

```
        Ball ball;
        Thread ballThread;
        int time, count;
        int mv=30;
        public BallJumpPan() {
            // 设置边框大小和颜色
            this.setPreferredSize(new Dimension(400, 300));
            this.setBackground(Color.black);
            this.setBorder(new LineBorder(Color.WHITE));
            // 设置小球的运动速度
            time=50;
            count=0;
            ball=new Ball(200, 150);
            // 启动运动线程
            ballThread=new Thread(this);
            ballThread.start();
        }
        public void paint(Graphics g) {
            super.paint(g);
            g.setColor(ball.color);
            g.fillOval(ball.xb, ball.yb, 20, 20);
        }
        // 线程启动后执行
        public void run() {
            while (true) {
                repaint();
                if (ball.xb>380 || ball.xb<0) {
                    ball.vx=-ball.vx;
                }
                if (ball.yb<0 || ball.yb>280) {
                    ball.vy=-ball.vy;
                }
                ball.xb+=ball.vx;
                ball.yb+=ball.vy;
                try {
                    Thread.sleep(time);
                } catch (InterruptedException e) {
                }
            }
        }

    }
}
```

(2) 保存程序后运行并调试程序，得到如图 7-4 所示的运行效果。

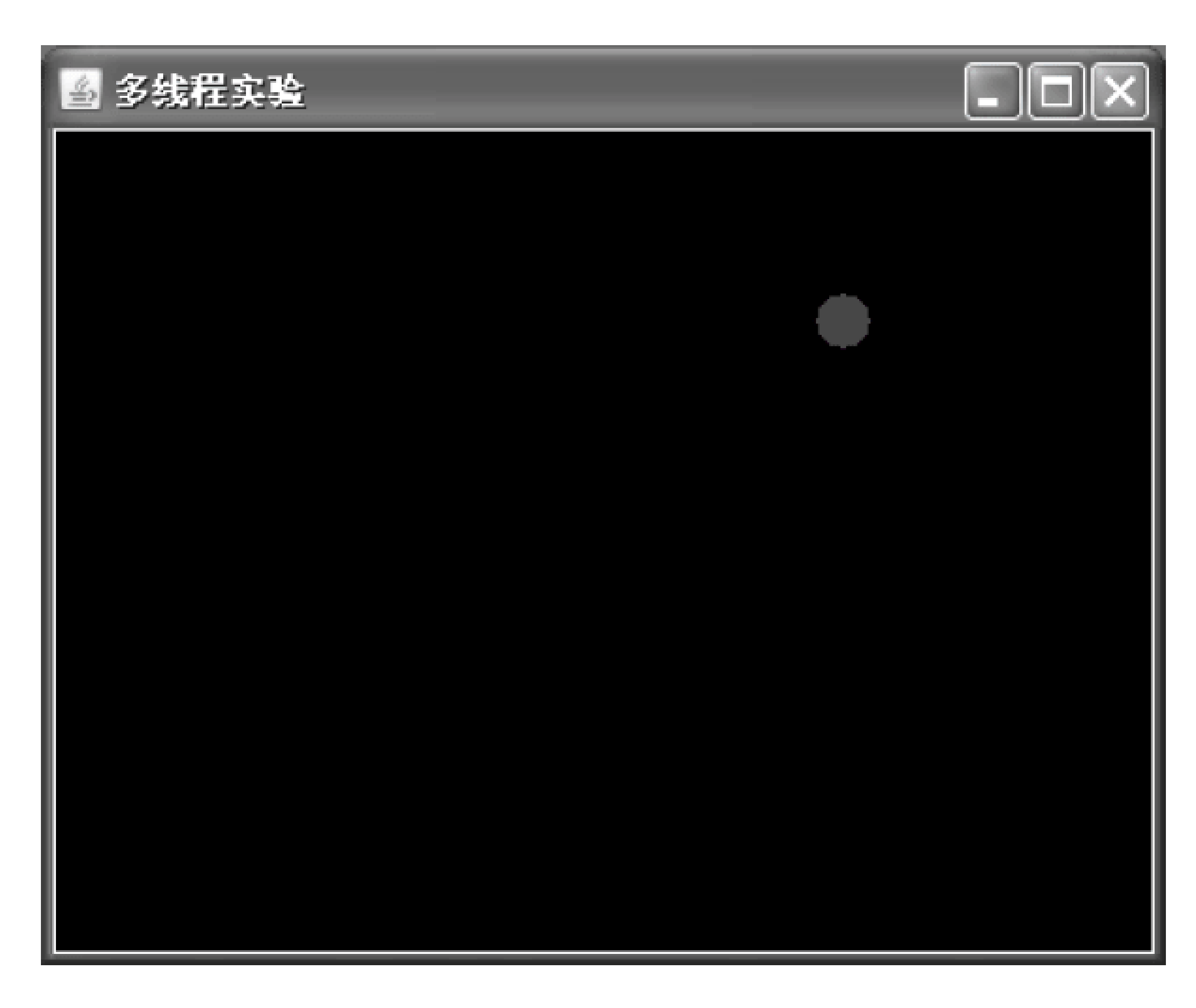


图 7-4　Java 多线程实验运行效果

练习 2：修改上面的程序，允许用户增加或减少新的球。最多提供 20 个球，为每个球随机选择颜色。

具体要求是：修改上述代码 Lab7_2_1.java，在其中加入一个按钮面板 btPanel，并在其中加入两个按钮 btAddBall、btReduceBall，分别用于实现运动小球的增加和减少，并控制小球个数最多 20 个，最少 0 个。新建一个 Java 文件 Lab7_2_2.java，结合 Lab7_2_1.java 将如下程序清单 7-5 输入并保存。仔细阅读程序清单 7-5，将程序清单 7-5 中的 代码 1 ～ 代码 8 补充完整，实现题意要求。

程序清单 7-5：

```
//Lab7_2_2.java
import java.awt.* ;
import java.awt.event.* ;
import javax.swing.* ;
import javax.swing.border.LineBorder;

public class Lab7_2_2 extends JFrame {
    private BallJumpPan ballPanel= 代码 1 ;
    //创建显示小球运动的面板
    private BtPan btPanel= 代码 2 ;
    //创建按钮面板

    public Lab7_2_2(String title) {
        super(title);
        this.getContentPane().add( 代码 3 , BorderLayout.CENTER);
        this.getContentPane().add( 代码 4 , BorderLayout.SOUTH);
```

```
        //将面板 ballPanel 和 btPanel 分别放置在显示窗口内
    }

    public static void main(String args[]) {
        Lab7_2_2 exe=new Lab7_2_2("多线程实验");
        exe.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        exe.pack();
        exe.setVisible(true);
    }

    private class Ball {
        int xb, yb;
        int vx, vy;
        boolean state=true;
        final int W=5, H=5;
        Color color;

        public Ball(int x, int y) {
            xb=x;
            yb=y;
            vx=-(int) (Math.random()*10+5);
            vy=-(int) (Math.random()*10+5);
            color=new Color((int) (Math.random()*255),
                    (int) (Math.random()*255), (int) (Math.random()*255));
        }
    }

    private class BallJumpPan extends JPanel implements Runnable {
        Ball[] ball;
        Thread ballThread;
        int time, count;
        int mv=30;

        public BallJumpPan() {
            // 设置边框大小和颜色
            this.setPreferredSize(new Dimension(400, 300));
            this.setBackground(Color.black);
            this.setBorder(new LineBorder(Color.WHITE));
            // 设置小球的运动速度
            time=50;
            count=0;
            ball=new Ball[20];
            ballThread = [代码 5];
```

```
        //创建新线程
        代码 6 ;
        //启动线程 ballThread
    }

    public int getCount() {
        return count;
    }

    public void paint(Graphics g) {
        super.paint(g);
        for (int i=0; i<count; i++) {
            g.setColor(ball[i].color);
            g.fillOval(ball[i].xb, ball[i].yb, 20, 20);
        }
    }

    public void run() {
        while (true) {
            代码 7 ;
            //重新绘制窗口
            for (int i=0; i<count; i++) {
                if (ball[i].xb>380 || ball[i].xb<0) {
                    ball[i].vx=-ball[i].vx;
                }
                if (ball[i].yb<0 || ball[i].yb>280) {
                    ball[i].vy=-ball[i].vy;
                }
                ball[i].xb +=ball[i].vx;
                ball[i].yb +=ball[i].vy;
            }
            try {
                代码 8 ;
                    //设置线程休眠时间为 time
            } catch (InterruptedException e) {
            }
        }
    }

    public void addBall() {
        if (count<20) {
            ball[count++]=new Ball(200, 150);
```

```
            }
        }

        public void reduceBall() {
            if (count>0) {
                ball[--count]=null;
            }
        }
    }

    private class BtPan extends JPanel {
        private JButton btAddBall, btReduceBall;
        private JLabel jlNumber;

        public BtPan() {
            btAddBall=new JButton("增加");
            btReduceBall=new JButton("减少");
            jlNumber=new JLabel(ballPanel.getCount()+"/20");
            this.setLayout(new GridLayout(1, 3, 10, 10));
            this.add(btAddBall);
            this.add(btReduceBall);
            this.add(jlNumber);
            if (ballPanel.getCount()==0)
                btReduceBall.setEnabled(false);
            if (ballPanel.getCount()==20)
                btAddBall.setEnabled(false);
            //动作监听
            btAddBall.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent arg0) {
                    ballPanel.addBall();
                    btReduceBall.setEnabled(true);
                    if (ballPanel.getCount()==20)
                        btAddBall.setEnabled(false);
                    jlNumber.setText(ballPanel.getCount()+"/20");
                }

            });
            btReduceBall.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent e) {
                    ballPanel.reduceBall();
                    btAddBall.setEnabled(true);
                    if (ballPanel.getCount()==0)
                        btReduceBall.setEnabled(false);
```

```
                    jlNumber.setText(ballPanel.getCount()+"/20");
                }
            });
        }

    }
}
```

填充并保存代码后，运行并调试程序，检查是否能得到如图 7-5 所示的运行效果。

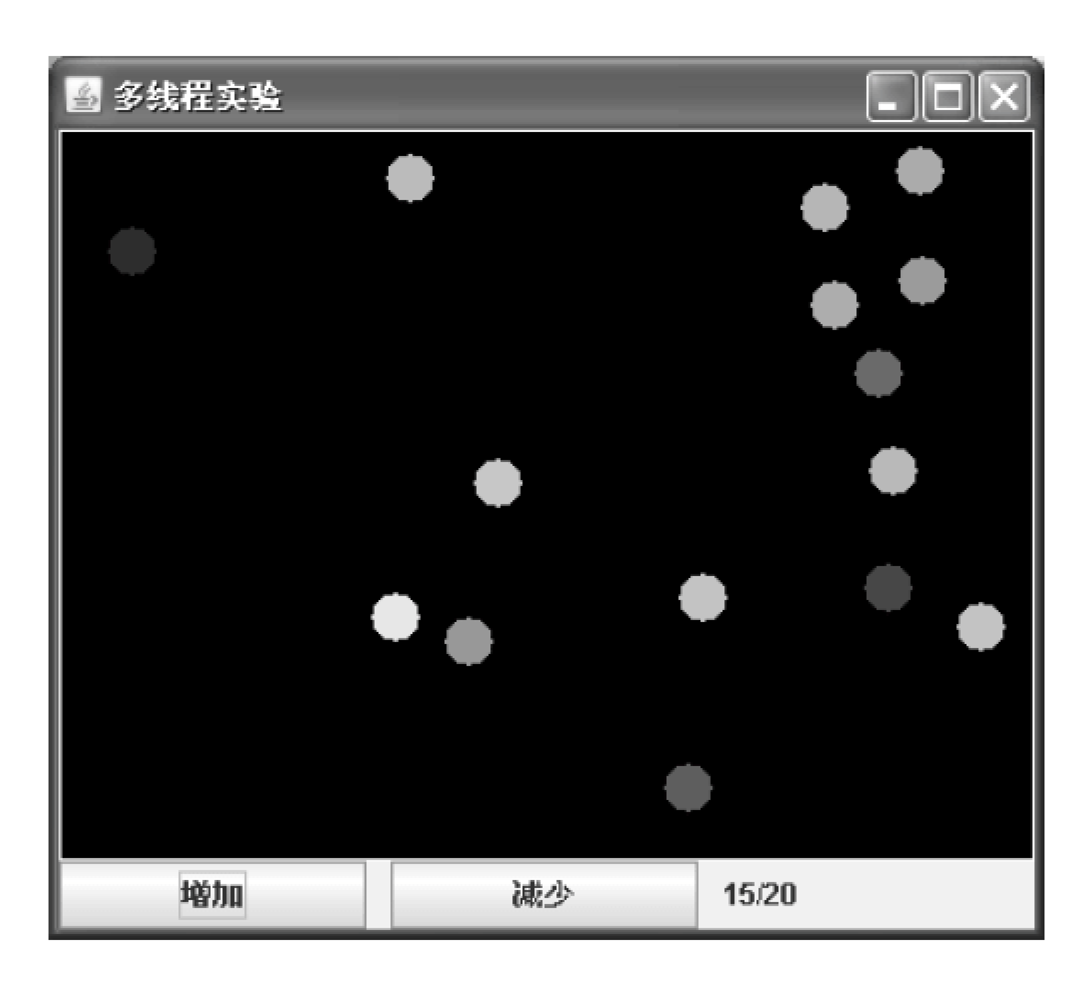


图 7-5　Java 多线程程序运行效果

第8章　JSP 简介

JSP(Java Server Pages)现在已经成为主流的 Web 程序开发技术之一,它具备了跨平台、通用性好、安全可靠等特点。在本章将概括介绍 JSP 的基本语法以及设计相关的实验,帮助读者开始使用 JSP 开发应用。

8.1　预备知识

8.1.1　JSP 的变量、方法与表达式

1. 变量和方法

JSP 中变量和方法的声明语法如下:

```
<%!声明语句 1;
    声明语句 2;
      ⋮
    声明语句 n
%>
```

2. 表达式

JSP 的表达式是由变量、常量和运算符组成的式子,它可以将计算结果转换成字符串直接在页面中输出。表达式的语法格式如下:

```
<%=表达式%>
```

8.1.2　JSP 注释元素

JSP 中的注释可以分为两类:一类是在客户端可见的注释,我们可以称其为 HTML 注释;还有一类是客户端不可见的注释,这类注释有隐藏注释以及脚本代码中的注释。

1. HTML 注释

其语法格式如下:

```
<!--注释语句<%=表达式%>-->
```

2. 隐藏注释

其语法格式如下:

```
<%--注释语句--%>
```

3. 脚本代码的注释

其语法格式如下:

```
<%//注释语句%>
```

或

```
<%/* 注释语句 */%>
```

8.1.3　JSP 指令元素

常用的 JSP 指令有 include 指令、page 指令和 taglib 指令。指令元素的语法格式如下：

```
<%@指令名 属性名 1="属性值 1", 属性名 2="属性值 2", … %>
```

1. include 指令

include 指令用来在当前 JSP 页面中加载需要插入的文本或代码文件，这些加载的代码将和原有的 JSP 代码合并成一个新的 JSP 文件，并由 JSP 引擎编译后执行。

其语法格式如下：

```
<%@include file="相对路径" %>
```

2. page 指令

page 指令用来定义整个 JSP 页面要使用的属性，如所使用的脚本语言、要导入的包文件、错误处理方法等等。

它的语法格式如下：

```
< %@  page
[language="java"]
[import="{package.class|package.* },..."]
[extends="package.class"]
[contentType="text/html;charset=ISO-8859-1" | "mimeType[;charset=CHARSET]"]
[session="True | False"]
[buffer="none | 8kb | size kb"]
[autoFlush="Ture | False"]
[isThreadSafe="True | False"]
[info="text"]
[errorPage="relativeURL"]
[isErrorPage="True | False"]
%>
```

3. taglib 指令

taglib 指令用于用户自定义标签库以及标记的前缀。

其语法格式如下：

```
<%@taglib uri="taglibURI" prifex="tagPrefix" %>
```

8.1.4　JSP 动作元素

常用的 JSP 动作包括<jsp:param>、<jsp:include>、<jsp:useBean>、<jsp:setProperty>、<jsp:getProperty>、<jsp:forward>和<jsp:plugin>等。

1. <jsp:param>动作

<jsp:param>动作用来为 JSP 页面的其他标记提供附加信息，通常它以“名/值”的方

式传递到页面中，其语法格式如下：

```
<jsp:param name="参数名" value="参数值"/>
```

2. <jsp:include>动作

<jsp:include>动作用来在生成的 Web 页面中插入指定的文件。

其语法格式如下：

```
<jsp:include page="文件名" flush="true"/>
```

或

```
<jsp:include page="文件名" flush="true">
<jsp:param name="参数名 1" value="参数值 1"/>
<jsp:param name="参数名 2" value="参数值 2"/>
     ⋮
</jsp:include>
```

3. <jsp:setProperty >动作、<jsp:getProperty>动作和<jsp:useBean>动作

(1) <jsp:setProperty >动作。<jsp:setProperty >动作是和<jsp:useBean>动作一起使用的，用来设置 Bean 的属性值，其语法格式如下：

```
<jsp:setProperty name="Bean 的名字" property="*"/>
```

或

```
<jsp:setProperty name="Bean 的名字" property="属性名" [param="参数名"]/>
```

或

```
<jsp:setProperty name="Bean 的名字" property="属性名" value="属性值"/>
```

(2) <jsp:getProperty >动作。<jsp:getProperty >动作也是和<jsp:useBean>动作一起使用，用来获取 Bean 中属性的值，所得到的值会转换成相应的字符串，然后发送到输出流输出。其语法格式如下：

```
<jsp:getProperty name="Bean 的名字" property="属性名"/>
```

(3) <jsp:useBean>动作。<jsp: useBean>动作用于加载要在 JSP 页面中使用的已经定义的 JavaBean，使用该动作在 JSP 页面中创建该 Bean 的实例，并指定它的名字以及作用范围。其语法格式如下：

```
<jsp:useBean id="name" scope="page|request|session|application " class="classname"/>
```

或

```
<jsp:useBean id="name" scope="page|request|session|application "
                        class="classname" type="typename"/>
```

或

```
<jsp:useBean id="name" scope="page|request|session|application "
```

```
            beanName="beanName" type="typename"/>
```

或

```
<jsp:useBean id="name" scope="page|request|session|application" type="typename"/>
```

4. <jsp:forward>动作

<jsp:forward>动作用于停止当前页面的剩余操作，而转向另一个 HTML 或 JSP 页面文件。客户端看到的仍然是当前页面的地址，但显示内容为转向的新页面。

其语法格式如下：

```
<jsp:forward page="目标文件 URL"/>
```

或

```
<jsp:forward page="目标文件 URL">
    <jsp:param name="参数名 1" value="参数值 1"/>
    <jsp:param name="参数名 2" value="参数值 2"/>
    ⋮
</jsp:forward>
```

5. <jsp:plugin >动作

<jsp:plugin>动作用于在客户端浏览器插入 Applet 程序或 JavaBean。

其语法格式如下：

```
<jsp:plugin type="bean | applet "
        code=" classFileName "
        codebase="classFileDirectoryName "
        name="instanceName "
        [archive="URIToAchive,… "]
        [align="bottom | top| middle| left| right "]
        [height="displayPixels "]
        [width=" displayPixels "]
        [hspace=" leftrightPixels "]
        [vspace=" topbottomPixels "]
        [jreversion=" JREVersionNumber | 1.1"]
        [nspluginurl=" URLToPlugin "]
        [iepluginurl=" URLToPlugin "]>
        [<jsp:params>
        [<jsp:param name="parameterName"
                value="parameterValue| <% =expression% >  "/>]
        </jsp:params > ]
        [<jsp:fallback>text message for user</jsp:fallback>]
        </jsp:plugin>
```

8.1.5 JSP 脚本元素

JSP 脚本(Scriptlet)是编写 JSP 代码时使用很频繁的元素，它通常是用 Java 编写的脚

本代码，可以定义变量和函数，也可以进行表达式求值、产生输出等操作。

Scriptlet 是介于“<%”和“%>”之间的 Java 代码段，可以在服务器端被编译执行，执行结果会嵌入页面文件并由服务器发回给客户端浏览器。

8.2 实验 8.1 配置 JSP 运行环境

实验目的：

(1) 熟悉 JSP 程序运行环境的配置方法。

(2) 了解 JSP 程序的运行原理。

实验内容：

本书主要以 JDK 6.0+Tomcat 6.0 作为运行 JSP 程序的实验环境，因此本次实验要求在 PC 上安装以上实验环境，并进行测试。

实验步骤：

1. 安装 JDK 6.0

(1) 将 JDK 6.0 的安装程序下载到本机上后，按照安装程序的安装向导可以直接进行安装。进入安装向导界面后可以进行 JDK 的路径设置，如图 8-1 所示。如果使用默认路径则直接单击“下一步”按钮，程序会自动进行安装。如果要自定义安装路径，则单击“更改”按钮，选择相应路径后再单击“下一步”按钮进行后续安装。

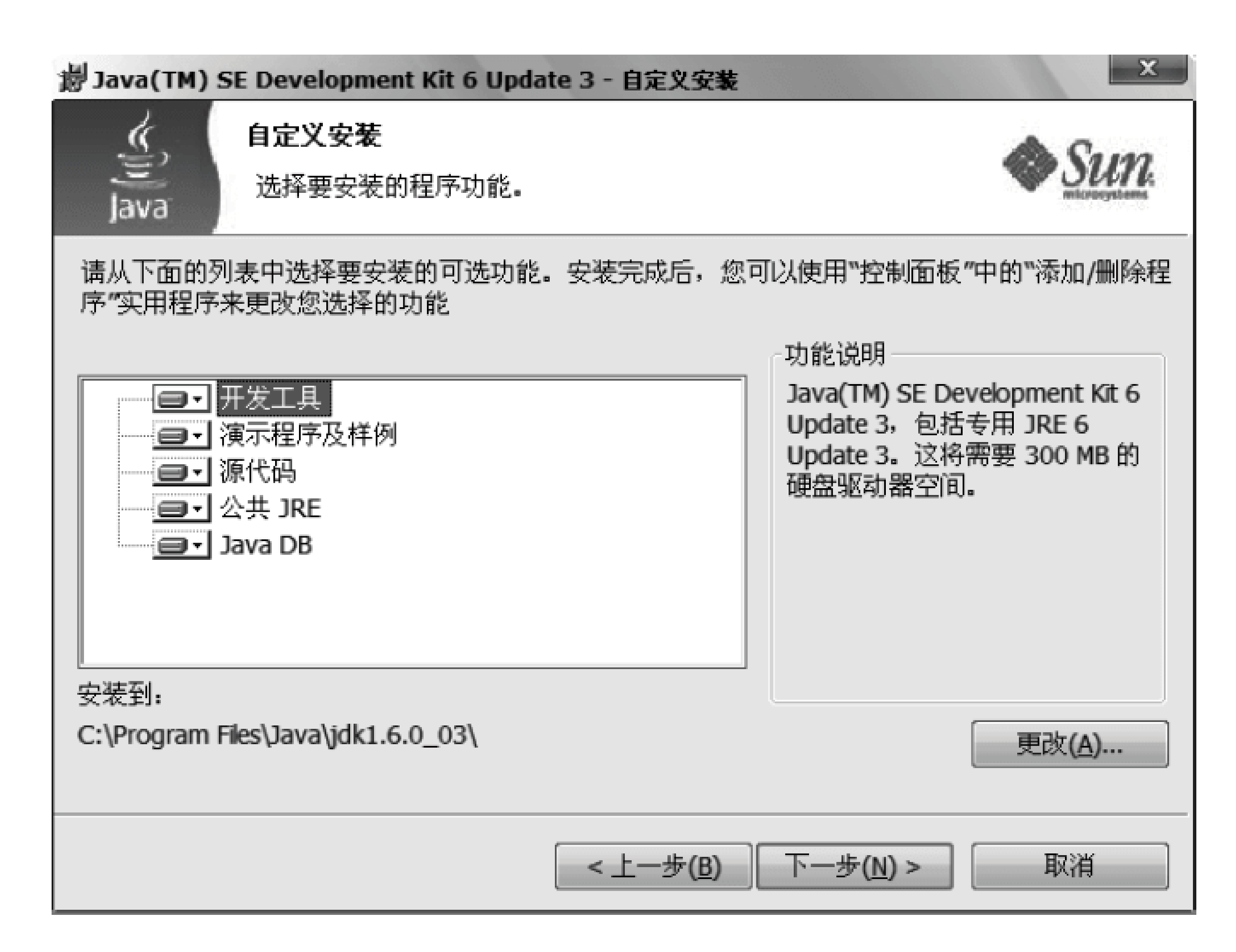


图 8-1 JDK 安装程序路径设置界面

(2) 安装好 JDK 6.0 后，如果想在系统的任意目录下编译和运行编写好的 Java 程序，需要先设置环境变量。本实验中将在 Windows 系统下环境变量中设置 JDK 运行路径。JDK 的安装路径均使用安装程序的默认路径：C:\Program Files\Java\jdk1.6.0_03。

① 右击“我的电脑”，在弹出的快捷菜单中选择“属性”命令，在弹出对话框中单击“高级”选项卡的“环境变量”按钮(如图 8-2 所示)。



图 8-2　Windows 系统属性界面

② 在“系统变量”列表中选择 Path 项，单击“编辑”按钮(如图 8-3 所示)，在弹出对话框中的“变量值”文本框中添加“；C:\Program Files\Java\jdk1.6.0_03\bin”(如果前一个变量值后已有“；”则将路径信息加入即可)，单击“确定”按钮，如图 8-4 所示。



图 8-3　Windows 系统环境变量操作界面

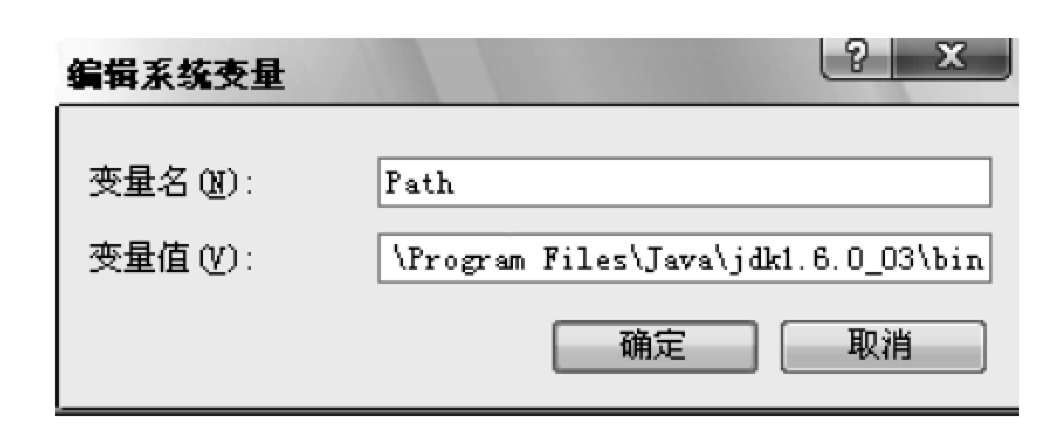


图 8-4　编辑系统变量 Path

③ 在“系统变量”栏下单击“新建”按钮(如图 8-5 所示)，在弹出对话框中的“变量名”输入框中输入“java_home”，“变量值”输入框中输入“C:\Program Files\Java\jdk1.6.0_03”，

单击“确定”按钮，如图 8-6 所示。

④ 在“系统变量”栏单击“新建”按钮，在弹出对话框中的“变量名”输入框中输入“classpath”，在“变量值”后输入“C:\Program Files\Java\jdk1.6.0_03\lib\tools.jar;C:\Program Files\Java\jdk1.6.0_03\jre\lib\rt.jar;.;”，单击“确定”按钮，如图 8-7 所示。至此环境变量设置完毕。



图 8-5　选择“新建”系统变量



图 8-6　新建系统变量 java_home

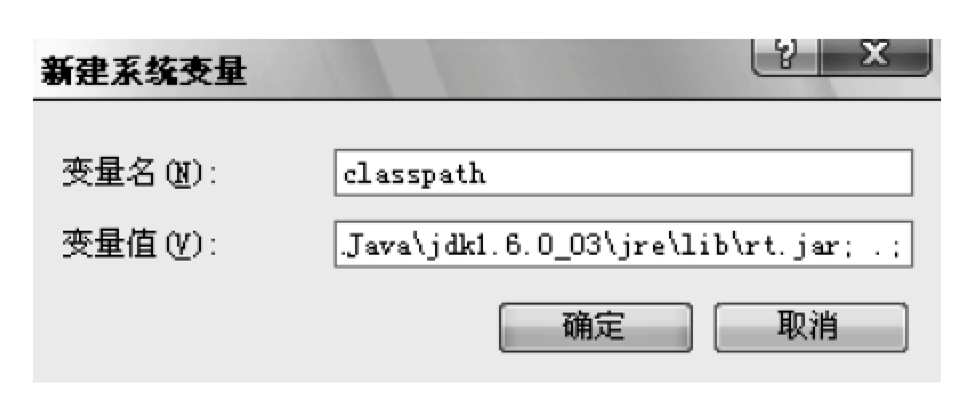


图 8-7　新建系统变量 classpath

至此环境变量设置完毕。可以通过在 DOS 环境下直接键入 javac 命令按回车键来检查环境变量是否设置成功，如果设置成功则可看到如图 8-8 所示结果。



图 8-8　javac 命令的相关使用语法说明

2．安装 Tomcat 6.0

将 Tomcat 6.0 的安装程序下载到本机上后，按照安装程序的安装向导可以直接进行安装。具体步骤如下。

(1) 运行安装程序，进入如图 8-9 所示的安装向导界面。

(2) 单击图 8-9 中的 Next 按钮，进入图 8-10 所示界面。单击 I Agree 按钮，进入组件



图 8-9　Tomcat 6.0 安装向导界面(一)

选择界面。

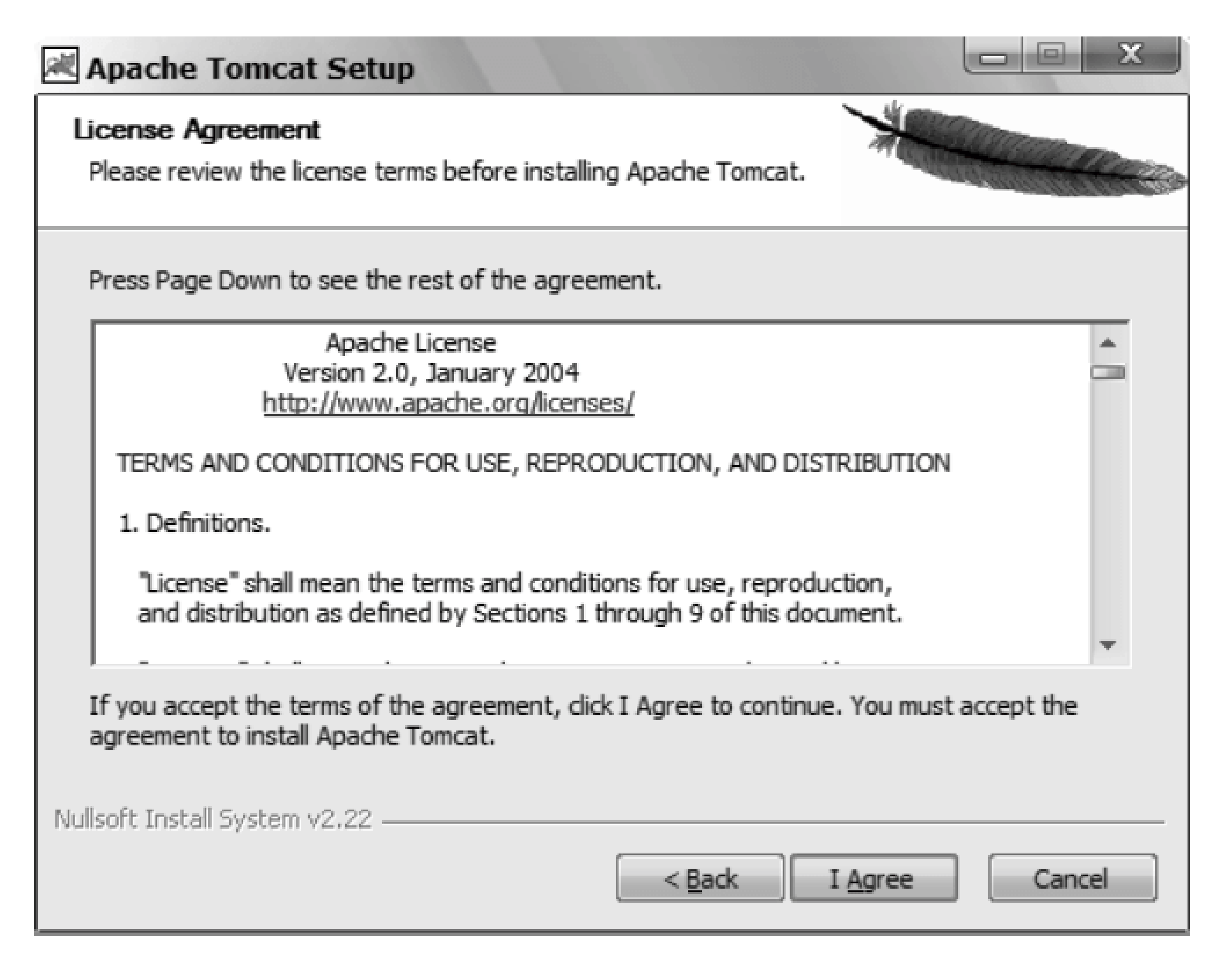


图 8-10　Tomcat 6.0 安装向导界面(二)

(3) 在图 8-11 所示对话框中可以选择你所需要的组件,通常可以全部选定,如图 8-11 所示,单击 Next 按钮。

(4) 选择安装路径。可以使用默认路径,也可以根据自身需要自定义安装路径。设置好后单击 Next 按钮,如图 8-12 所示。

(5) 设置 Tomcat 基本参数。可以在此设置 Tomcat 的连接端口(默认为 8080)、管理员登录时使用的用户名和密码。如果设置了用户名和密码,则在每次登录 Tomcat 环境时都需要输入设定的用户名和密码,才能够进入 Tomcat 环境。设置完成后单击 Next 按钮,如图 8-13 所示。

(6) 设置 Java 虚拟机的路径。在安装 Tomcat 前如果已经安装好 JDK 环境后,安装向导会自动查找到 Java 虚拟机的路径,你也可以根据需要修改成其他已经安装好的 Java 虚

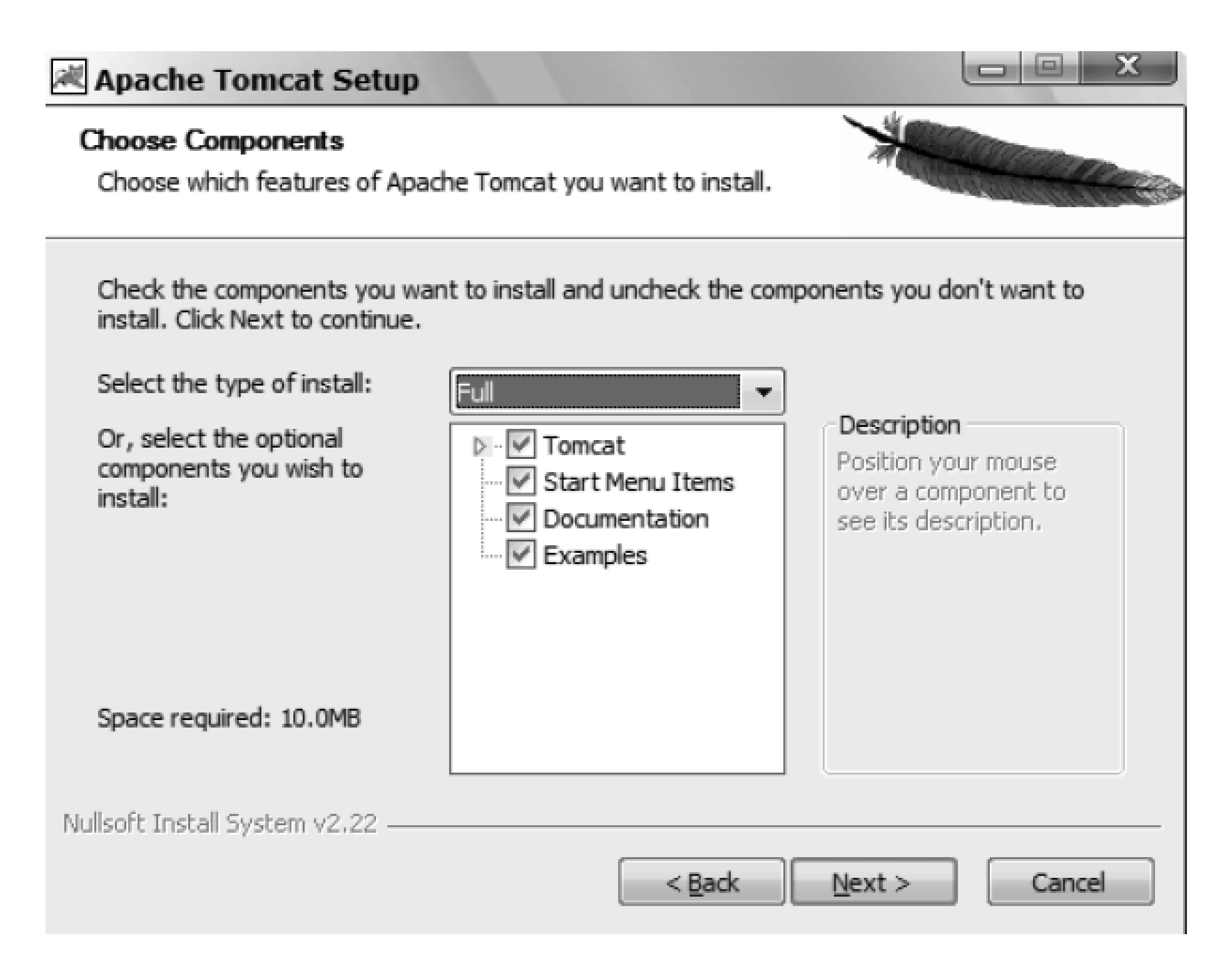


图 8-11　Tomcat 6.0 安装向导界面(三)

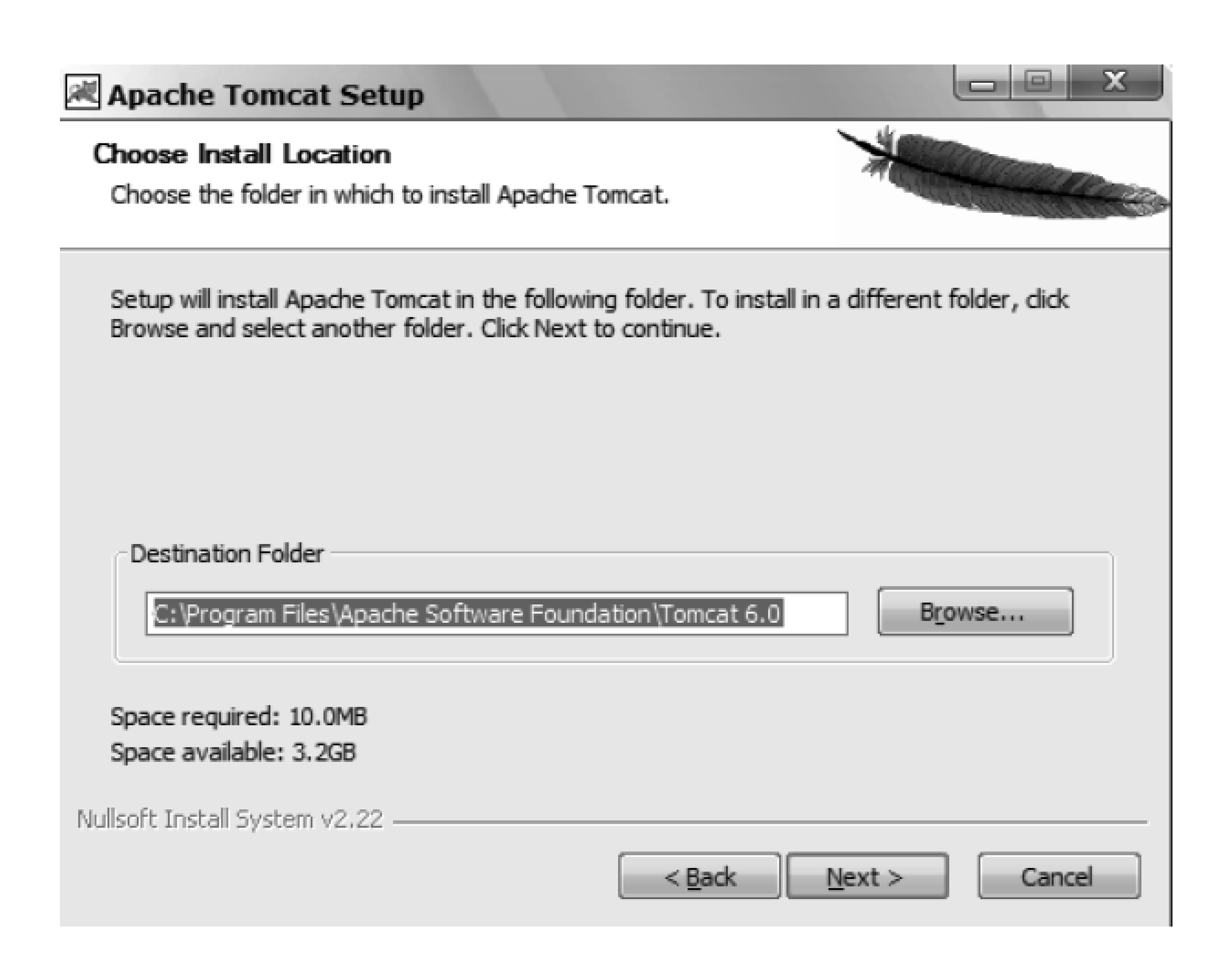


图 8-12　Tomcat 6.0 安装向导界面(四)

图 8-13　Tomcat 6.0 安装向导界面(五)

拟机路径。设置好后单击 Install 按钮，安装程序将按照之前设置的所有参数安装 Tomcat。安装完成将看到如图 8-14 的界面。

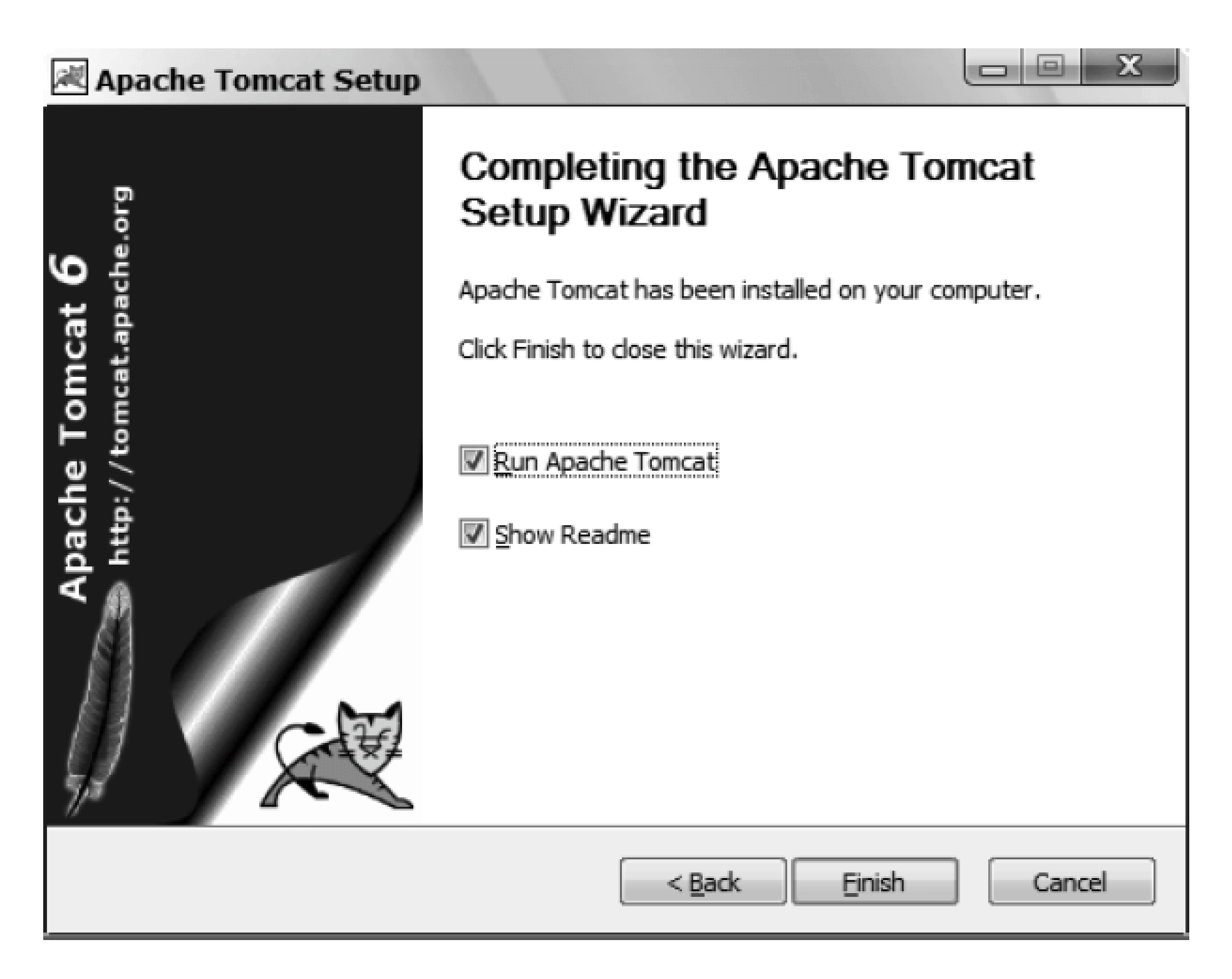


图 8-14　Tomcat 6.0 安装向导界面(六)

(7) 测试 Tomcat 环境是否成功安装。可以启动 Tomcat，打开浏览器，在地址栏中输入 "http://localhost:8080"，当浏览器中显示如图 8-15 的页面时，说明 Tomcat 已经安装成功。

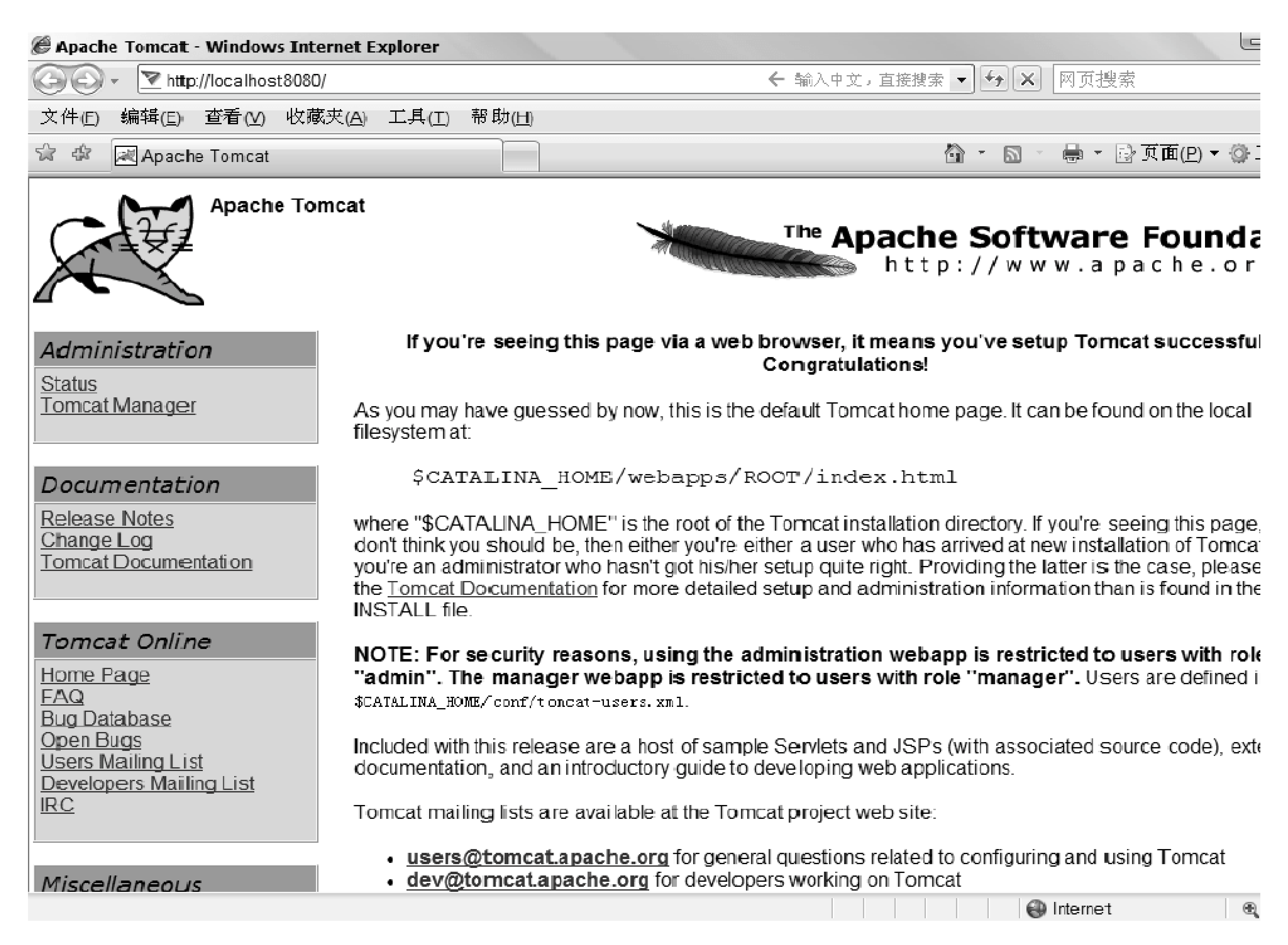


图 8-15　Tomcat 服务器环境主页面

8.3　实验 8.2　网上购物订单

实验目的：

（1）熟悉 JSP 程序的运行。

（2）掌握 JSP 程序基本语法。

实验内容：

当用户在网上选购相应货品后，需要填写相应订单来提供收货人信息，以便于厂商邮寄货品。本实验要求编写一个简单的 JSP 程序实现该购物订单的基本功能：将客户在 Shopping.html 页面中输入的联系方式和邮寄地址等信息内容读取出来并显示在另一页面 Shopping.jsp 中。

实验步骤：

（1）新建一个名为 Shopping.html 的 HTML 页面，在其中输入程序清单 8-1 的代码，其页面内容如图 8-16 所示。当用户在其中输入用户的相关信息后，单击“确定”按钮，则跳转至 Shopping.jsp 页面，源代码见程序清单 8-2，该页面将用户的输入信息顺序显示，如图 8-17 所示。请按实验结果将程序清单 8-1 和 8-2 的空白处 代码 1 ～ 代码 9 补充完整。

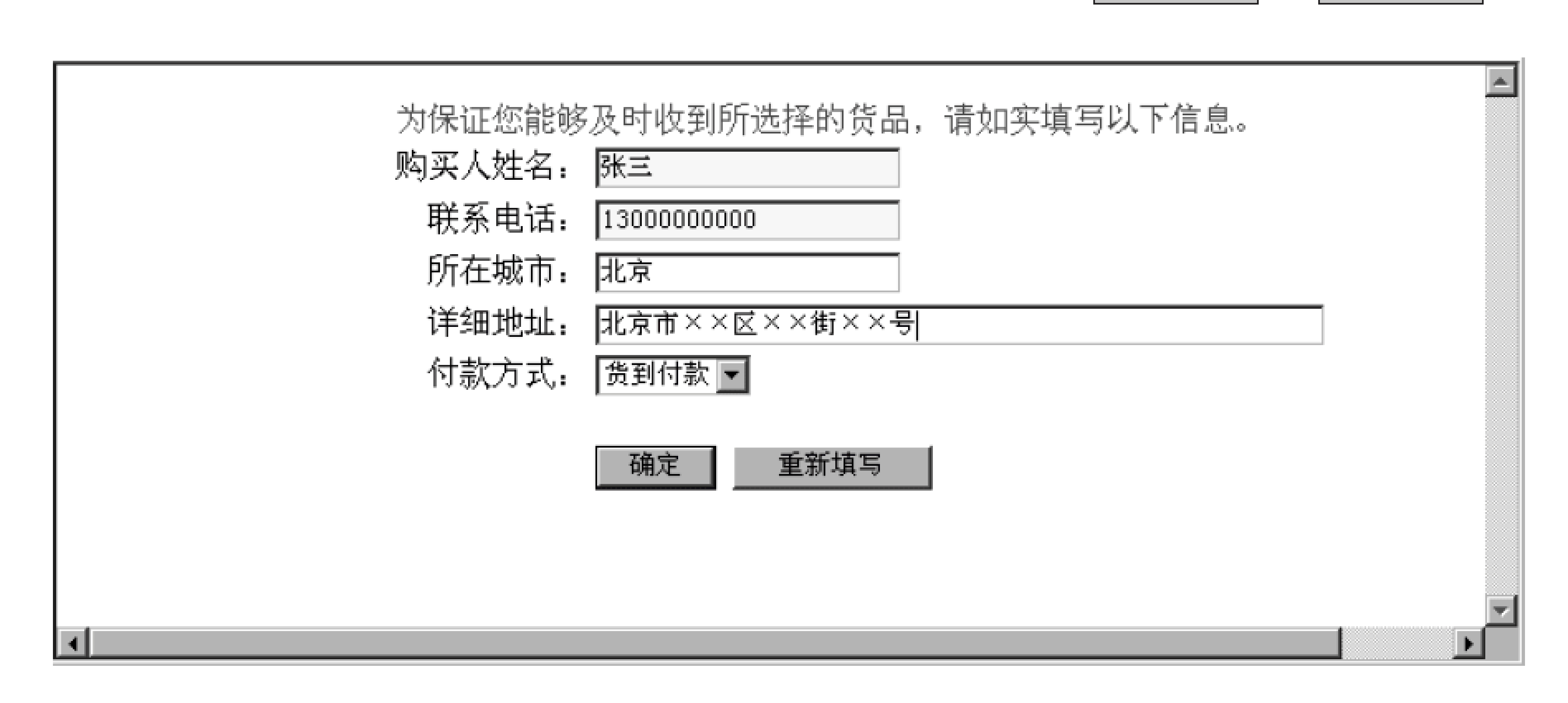


图 8-16　实验 8.2 运行结果（一）

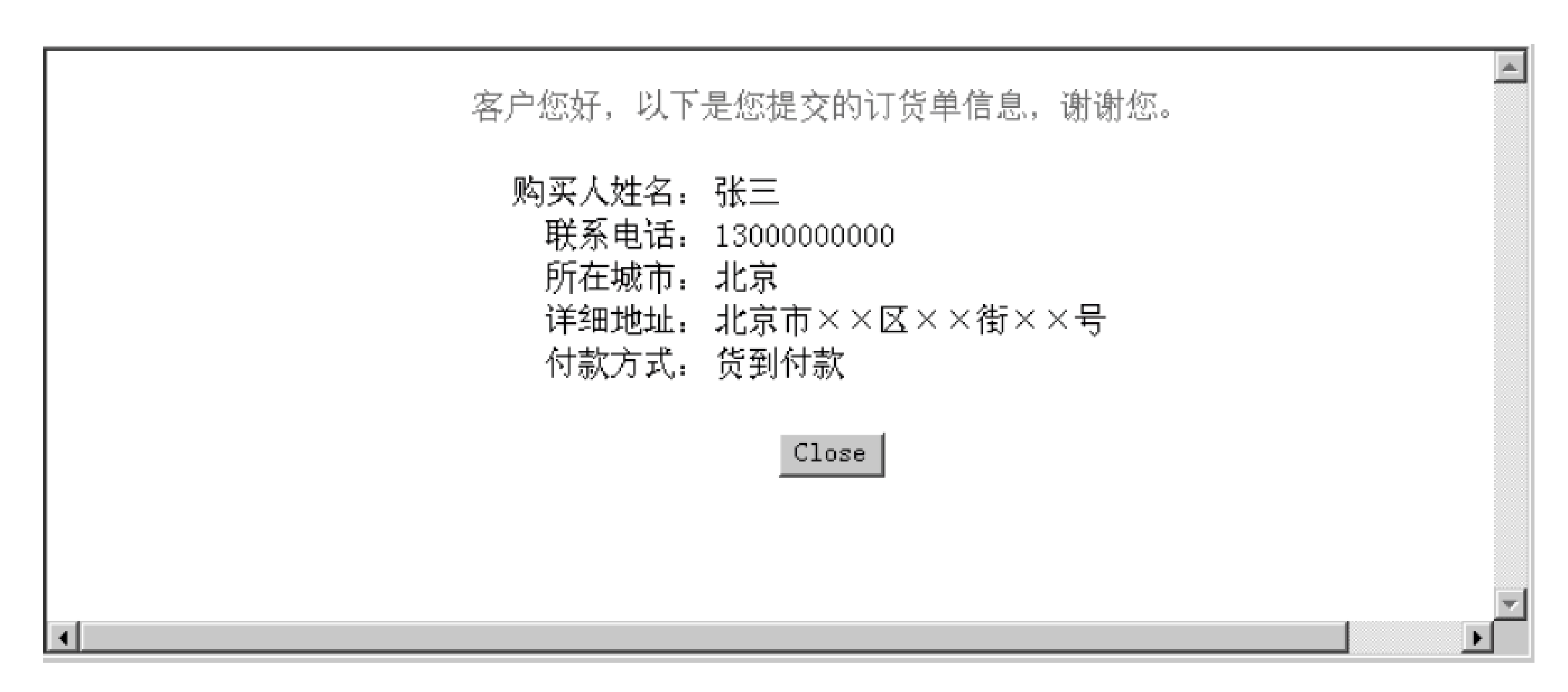


图 8-17　实验 8.2 运行结果（二）

程序清单 8-1：

```
<!--Shopping.html 源代码-->
<html>
<head>
<script language="javascript">
<!--
function CheckSubmit()
{
  if( 代码 1 ==""){
      alert("请输入购买人姓名!");
      document.orderform.name.focus();
      return false;
      }
  if( 代码 2 ==""){
      alert("请输入联系电话!");
      document.orderform.phonenumber.focus();
      return false;
}
  if( 代码 3 ==""){
      alert("请输入购买人所在城市!");
      document.orderform.city.focus();
      return false;
}
  if( 代码 4 ==""){
      alert("请输入详细地址!");
      document.orderform.address.focus();
      return false;
}
  return true;
}
</script>
</head>

<body>
<div align="center">
<table width="750" border="0" cellspacing="1" cellpadding="1">
  <tr>
    <td ><div align="center">
      <table width="80%" border="0" cellpadding="1" cellspacing="1" class="td">
        <form name="orderform" action="Shopping.jsp" method="post">
          <tr align="center">
            <td colspan="2">
              <font color="red">
```

```
        为保证您能够及时收到所选择的货品，请如实填写以下信息。
        </font>
      </td>
    </tr>
    <tr>
      <td width="30%"align="right">购买人姓名：</td>
      <td align="left">
      <input name="name" type="text" size="20">
  </td>
    </tr>
    <tr>
      <td align="right">联系电话：</td>
      <td align="left">
        <input name="phonenumber" type="text" size="20">
      </td>
    </tr>
    <tr>
      <td align="right">所在城市：</td>
      <td align="left">
          <input name="city" type="text" size="20">
      </td>
    </tr>
    <tr>
      <td align="right">详细地址：</td>
      <td align="left"><input name="address" type="text" size="50"></td>
    </tr>
    <tr>
      <td align="right">付款方式：</td>
      <td align="left">
         <select name="paying">
           <option value="1">邮政汇款</option>
           <option value="2" selected="selected">货到付款</option>
         </select>
      </td>
    </tr>
    <tr>
      <td colspan="2" align="center">
         
      </td>
    </tr>
    <tr>
      <td align="right"> </td>
      <td align="left">
        <input name="submit" type="submit" value=" 确定 "
                  onClick="return CheckSubmit();">
```

```
                <input name="reset" type="reset" value=" 重新填写 ">
              </td>
            </tr>
          </form>
        </table>
      </div>
      </td>
    </tr>
    </table>
</div>
</body>
</html>
```

程序清单 8-2：

```
<!--Shopping.jsp 源代码-->
<%@page contentType="text/html; charset=GB2312"%>
<%
    request.setCharacterEncoding("gb2312");
%>
<html>
<head>
<script language="JavaScript">
<!--
    function closer(){
         window.close();
    }
-->
</script>
</head>
<body>
<div align="center">
<%
     String name =request.getParameter("name");
     //获取输入的购买人姓名信息
     String phonenumber =request.getParameter("phonenumber");
     //获取输入的联系电话信息
     String city =request.getParameter("city");
     //获取输入的购买人所在城市信息
     String address =request.getParameter("address");
     //获取输入的购买人详细地址信息
     String paying =request.getParameter("paying");
     //获取输入的支付方式信息
%>

<table width="750" border="0" cellspacing="1" cellpadding="1">
```

```
<tr>
    <td ><div align="center">
        <table width="80%" border="0" cellpadding="1" cellspacing="1" class="td">
            <tr>
                <td colspan="2" align="center">
                    <font color="red">客户您好,以下是您提交的订货单信息,谢谢您。</font>
                </td>
            </tr>
            <tr>
                <td colspan="2" align="center">
                  
                </td>
            </tr>
            <tr>
                <td width="40%"align="right">购买人姓名:</td>
                <td align="left">
                 代码 5
                </td>
            </tr>
            <tr>
                <td align="right">联系电话:</td>
                <td align="left">
                 代码 6
                </td>
            </tr>
            <tr>
                <td align="right">所在城市:</td>
                <td align="left">
                 代码 7
                </td>
            </tr>
            <tr>
                <td align="right">详细地址:</td>
                <td align="left">
                 代码 8
                </td>
            </tr>
            <tr>
                <td align="right">付款方式:</td>
                <td align="left">
                 代码 9
                </td>
            </tr>
            <tr>
```

```
                <td colspan="2" align="center">
                  
                </td>
            </tr>
            <tr>
                <td colspan="2" align="center">
                 <input type=button value="Close" onClick="closer()">
                </td>
            </tr>
        </table>
    </div>
    </td>
</tr>
</table>
</div>
</body>
</html>
```

（2）在 Tomcat 安装目录下的 webapps 目录下新建一个文件夹(名称可设定为 myjsp)，将编写好的 Shopping. html 文件和 Shopping. jsp 文件放置到该文件夹中。在本实验中按照所设置的 Tomcat 环境，将这两个文件放到 C:\Program Files\Apache Software Foundation\Tomcat 6.0\webapps\myjsp 目录下。

（3）启动 Tomcat，打开浏览器，在 URL 中输入"http://localhost:8080/myjsp/Shopping. html"，观察是否显示图 8-16 所示页面，在按图 8-15 在页面中输入相关信息后，单击"确定"按钮，观察是否跳转至图 8-17 所示页面。

说明： 本实验中在 Shopping. jsp 文件中使用了 JSP 的内置对象 request 的 getParameter()方法来获取从前一页面(Shopping. html 文件)中传递过来的表单内数据信息。该方法是 request 对象的一个常用方法，它的方法原型为：String getParameter(String name)；用来取得从客户端传递给服务器端的参数值，具体参数由参数 name 确定。

8.4 实验 8.3 简单购物车的实现

实验目的：

（1）进一步掌握 JSP 基本语法。
（2）熟悉 JSP 相关指令元素和动作元素。

实验内容：

编写 JSP 程序实现购物车的基本功能，即向购物车中添加或删除商品，并给出购物车中的商品信息和总金额。

实验步骤：

新建一个名为 Cart. html 的 HTML 文件，在其中输入程序清单 8-3 所示的源代码，该

页面运行效果如图 8-18 所示。在该页面中选择所购商品后，将转至 ShowCartInf.jsp 的页面（源代码见程序清单 8-4），用来显示购物车中所有的商品信息、每件商品的数量以及总价格，运行效果如图 8-19 所示。若在 ShowCartInf.jsp 页面单击“继续购物”按钮则回到 Cart.html 页面继续选择所需商品，如图 8-20 所示；若结束购买则可单击“完成”按钮。如果要删除购物车中的某种商品，可在 ShowCartInf.jsp 的页面中显示的相应商品右部单击“删除”按钮，则将转至 Delete.jsp 页面（源代码见程序清单 8-5）删除该商品的所有信息，然后从 Delete.jsp 页面自动跳转回到 ShowCartInf.jsp 页面重新显示删除所选商品后购物车的信息，如图 8-22 所示。实验中用到的 JavaBean 见程序清单 8-6。请根据所给程序代码及实验结果将 ShowCartInf.jsp 和 Delete.jsp 中空白部分 代码 1 ～ 代码 7 补充完整。

图 8-18　实验 8.3 运行结果(一)

图 8-19　实验 8.3 运行结果(二)

图 8-20　实验 8.3 运行结果(三)

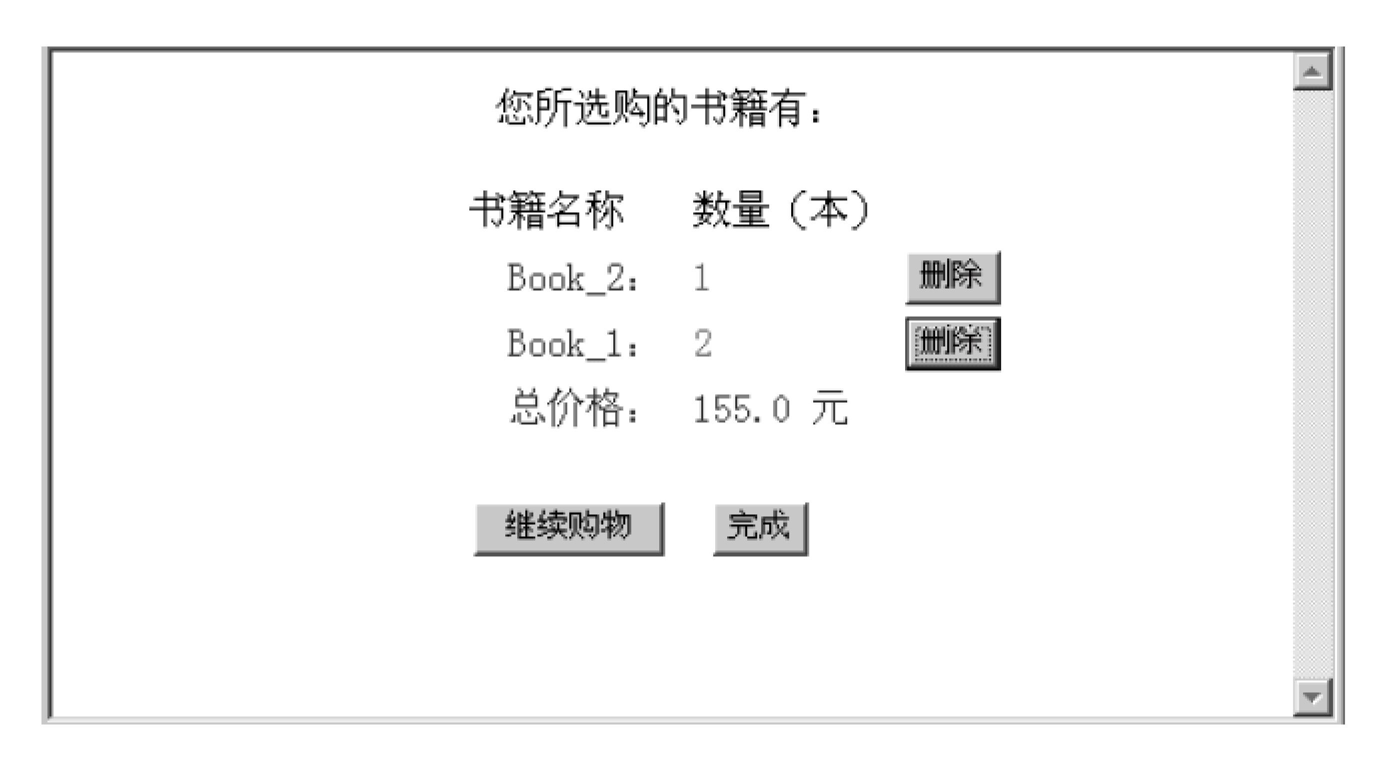


图 8-21　实验 8.3 运行结果(四)

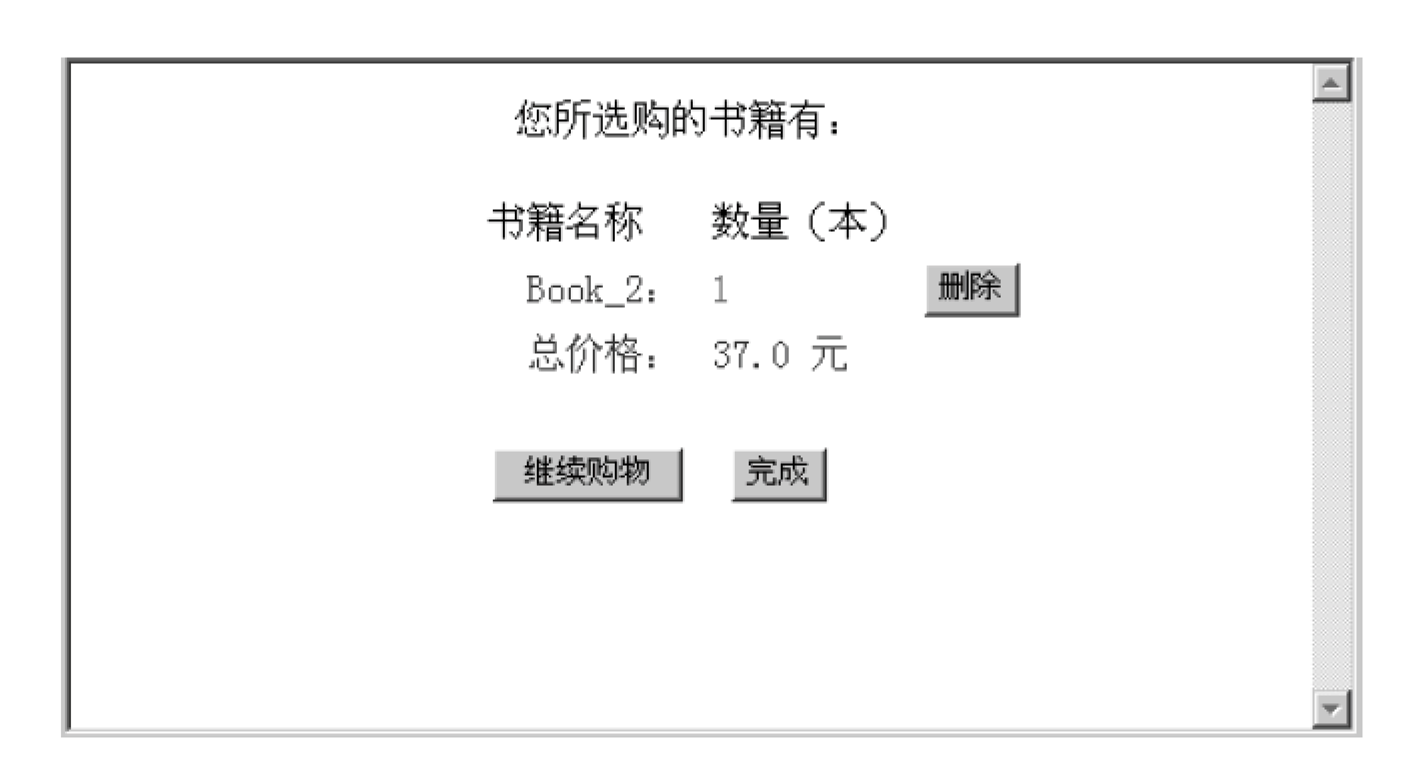


图 8-22　实验 8.3 运行结果(五)

程序清单 8-3：

```
<!--Cart.html 源代码-->
<html>
<head>
    <title>购物车</title>
</head>
<body>
<div align="center">
<form name="Nform" method="post" action="ShowCartInf.jsp" >
<table width="80%" border="0">
<tr>
<td width="50%" height="30" align="right">
    请选择您要购买的书籍：
</td>
<td width="50%" align="left"> 
<select name="BName">
     <option value="Book_1" selected>Book_1</option>
     <option value="Book_2">Book_2</option>
     <option value="Book_3">Book_3</option>
     <option value="Book_4">Book_4</option>
     <option value="Book_5">Book_5</option>
     <option value="Book_6">Book_6</option>
```

```
</select></td>
</tr>

<tr>
<td width="50%" height="30" align="right">
    购买数量：
</td>
<td width="50%" align="left"> 
<input type="text" name="BNumber" value="1" size="5"></td>
</tr>
</table>
<p>
<input type="submit" name="s" value="提交">  
<input type="reset" name="r" value="重新挑选">
</p>
</form>
</div>
</body>
</html>
```

程序清单 8-4：

```
<!--ShowCartInf.jsp 源代码-->
<%@ 代码 1 contentType="text/html; charset=gb2312" language="java"%>
<%@ 代码 2 import="java.util.*"%>
<html>
<head>
    <title>购物车</title>
</head>

<jsp:代码 3 id="Cart" scope="session" class="Cart.cart"/>
<jsp:代码 4 name="Cart" property="*"/>
<%
String bookName=request.getParameter("BName");
//获取书名
String bookNumber=request.getParameter("BNumber");
//获取书的数量
if(bookName!=null && bookName !="")
{
    int nbookNumber=Integer.parseInt(bookNumber);
    Cart.add(bookName,nbookNumber);
}
Hashtable h=Cart.ShowCartInf();
Enumeration e=h.keys();
float sum=Cart.getSum();
```

```
%>
<body>
<div align="center">
    <p>您所选购的书籍有：</p>
<table width="80%" border="0">
<tr>
    <td width="50%" height="25" align="right">
        书籍名称   
    </td>
<td width="50%" align="left" colspan="2">
 数量(本)
</td>
     </tr>
<%
    //显示购物车中所有商品信息
     while(e.hasMoreElements())
     {
     String name =e.nextElement().toString();
     int number = ((Integer)h.get(name)).intValue();
%>
     <tr>
     <td width="50%" height="25" align="right">
          <font color="#0000FF"> 代码 5 ：</font>
     </td>
     <td width="20%" align="left"> <font color="#FF0000"> 代码 6 </font></td>
     <td width="30%" align="left"> 
          <input type="button" name="BName" value="删除"
          onClick="javascript:window.location='Delete.jsp? BName=<%=name%>'">
</td>
</tr>

<%
    }

%>
<tr>
<td width="50%" height="25" align="right">
      <font color="#0000FF">总价格：</font>
</td>
<td width="50%" height="25" align="left" colspan="2">
      <font color="#0000FF">  代码 7 nbsp;元</font>
</td>
</tr>
    <tr>
```

```
        <td colspan="3"> </td>
        </tr>
        <tr>
        <td width="50%" height="25" align="right">
          <input type="button" name="continue" value="继续购物"
              onClick="javascript:window.location='Cart.html'">
        </td>
      <td width="50%" height="25" align="left" colspan="2">
            <input type="button" name="finish" value="完成"
          onClick="javascript:window.location='#'">
</td>
      </tr>
</table>
</div>
</body>
</html>
```

程序清单 8-5:

```
<!--Delete.jsp 源代码-->
<%@ 代码 1  contentType="text/html; charset=gb2312" language="java"%>
<html>
<head>
    <title>购物车</title>
</head>

<jsp: 代码 3  id="Cart" scope="session" class="Cart.cart"/>
<jsp: 代码 4  name="Cart" property="*"/>

<body>
<div align="center">
<%
String bookName=request.getParameter("BName");
Cart.delete(bookName);
response.sendRedirect("ShowCartInf.jsp");
%>
</div>
</body>
</html>
```

程序清单 8-6:

```
//cart.java 源代码
package Cart;
import java.util.*;
import java.io.*;
```

```
public class cart implements Serializable
{
float Sum=0;
Hashtable<String,Integer>Goods=new Hashtable<String,Integer>();

public void cart()
{
}

//获取每件商品的价格
public float getPrice(String goodsname)
{
    if(goodsname.equals("Book_1"))
        return new Float(59.0);
    else if(goodsname.equals("Book_2"))
        return new Float(37.0);
    else if(goodsname.equals("Book_3"))
        return new Float(49.0);
    else if(goodsname.equals("Book_4"))
        return new Float(27.0);
    else if(goodsname.equals("Book_5"))
        return new Float(32.0);
    else if(goodsname.equals("Book_6"))
        return new Float(28.0);
    else
        return new Float(0);
}

//将某个商品信息加入购物车
public void add(String GoodsName, int GoodsNumber)
{
if(Goods.containsKey(GoodsName))
{
    int Num=((Integer)Goods.get(GoodsName)).intValue();
    Num=Num+GoodsNumber;
    Goods.put(GoodsName,new Integer(Num));
    Sum=Sum+getPrice(GoodsName) * GoodsNumber;
}
else
{
    Goods.put(GoodsName,new Integer(GoodsNumber));
    Sum=Sum+getPrice(GoodsName) * GoodsNumber;
}
}
```

```
//获取购物车中所有商品信息
public Hashtable ShowCartInf()
{
return Goods;
}

//从购物车中删除一件商品信息
public void delete(String GoodsName)
{
    int GoodsNum=((Integer)Goods.get(GoodsName)).intValue();
    Goods.remove(GoodsName);
    Sum=Sum-getPrice(GoodsName)*GoodsNum;
}

//获取购物车内商品总价格
public float getSum()
{
    return Sum;
}
}
```

第 9 章　JSP 的内置对象

JSP 的内置对象是不需要声明就可以直接使用的特殊对象。这些特殊对象包含了特定的信息，它们的存在在一定程度上简化了 JSP 页面的开发，也提高了开发的便利性。在本章将对常见的内置对象进行介绍，并通过实验来具体了解这些常见内置对象的应用和作用范围。

9.1　预备知识

JSP 的内置对象有 9 种：out 对象、request 对象、response 对象、session 对象、application 对象、exception 对象、config 对象、page 对象和 pageContext 对象。表 9-1 简单描述了这 9 种内置对象的基本情况，对于这 9 种内置对象的详细介绍可以参看《Web 开发技术实用教程》。

表 9-1　常见的内置对象

对　象	所　属　类	描　述	作用范围
out	javax.servlet.jsp.JspWriter	输出流对象	page
request	javax.servlet.ServletRequest 的子类	触发 JSP 文件的请求对象	request
response	javax.servlet.ServletResponse 的子类	返回客户的响应对象	page
session	javax.servlet.http.HttpSession	用户的会话对象	session
application	javax.servlet.ServletContext	JSP 页面的应用上下文对象	application
pageContext	javax.servlet.jsp.PageContext	JSP 页面的上下文对象	page
config	javax.servlet.ServletConfig	初始化 JSP Servlet 的对象	page
page	java.lang.Object	JSP 页面 servlet 的当前请求处理实例	page
exception	java.lang.Throwable	访问错误页面产生的异常对象	page

1. 内置对象的分类

对于表 9-1 列出的 9 种内置对象从应用角度进行分类，可以分成 4 类：

(1) 与 I/O 有关的内置对象为 out、request 和 response。

(2) 与 JSP 执行时，提供有关上下文的内置对象为 seesion、application 和 pageContext。

(3) 与 Servlet 有关的内置对象为 page 和 config。

(4) 与 Error 错误有关的内置对象为 exception。

注意： *在这 9 种内置对象中，pageContext、request、response、session、application 和 pageContext 必须熟练掌握。*

2. 内置对象的作用域

内置对象的作用域表示内置对象的作用范围。内置对象有 4 种作用域：page、request、session 和 application。

(1) page 作用域表示具有这样作用域的内置对象只能在当前的 JSP 页面中有效。

(2) request 作用域是指当前请求中有效,当服务器响应了这次客户端的请求,再次请求时,原有的属性将会失去效用。

(3) session 作用域表示在客户端和服务器端的会话过程中有效,除非会话关闭或者会话过期;不同客户对应不同的 session。

(4) application 作用域表示在整个 Web 服务器的应用程序中发生作用,为所有的用户服务,即所有的客户共享 application 作用域的 web 应用。

9.2 实验 9.1 out 对象的输出处理

实验目的:

(1) 了解内置对象的基本概念。

(2) 了解 out 对象的常见方法,并应用这些常见方法解决实际问题。

(3) 进一步了解服务器端的 Web 应用。

实验内容:

(1) 用 out 对象设计并输出一个 HTML 二维表。该二维表输出 out 对象的常见方法和方法说明。

(2) 用 out 对象设计输出一个 XHTML 网页,实现菜单功能。

实验步骤:

练习 1:输出 HTML 网页的二维表。

本次练习是输出一个关于"out 对象的常见方法"的二维表,该二维表由 3 列(方法名、方法说明、示例),11 行构成。11 行包括表头和 out 对象常使用的方法 clear()、clearBuffer()、close()、flush()、isAutoFlush()、getBufferSize()、gerRemaining()、newLine()、print()及 println()。注意比较方法 clear()、clearBuffer()、close()和 flush()的不同之处。要求仔细阅读下列程序清单 9-1 中列出的 outDemo.jsp,找出并修改程序中存在的错误,使得运行结果类似图 9-1。并回答下列问题。

程序清单 9-1:

```
<!--outDemo.jsp-->
<%@page contentType="text/html; charset=gb2312" language="java"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>out 对象常见的方法</title>
<style type="text/css">
caption{
```



方法	说明	示例
clear()	清除缓冲区的内容	
clearBuffer()	清除缓冲区的当前内容	
close()	关闭输出流	
flush()	强制输出缓冲区的数据	
isAutoFlush()	判断缓冲区是否具有强制输出autoFlush的功能	true
getBufferSize()	返回缓冲区的的大小	8192
getRemaining()	返回缓冲区没有占用的空间	7141
newLine()	输出一新行	
print()	输出信息	输出文字
println()	换行输出信息	换行输出一行

图 9-1　out 对象常见方法的输出表

```
        color:#FF6600;
        }
th{
        background:#FFFF00;
        color:#0000FF;
        }
td{
         color:#000099;
        }
</style>
</head>

<body>
<table border="1">
<caption>out 对象的常见方法</caption>
<tr>
    <th>方法</th>
    <th>说明</th>
    <th>示例</th>
</tr>
<%
   out.println("<tr>");
   out.println("<td>clear()</td>");
   out.println("<td>清除缓冲区的内容</td>");
   out.println("<td> </td>");
   out.println("</tr>");

   out.println("<tr>");
```

```
out.println("<td>clearBuffer()</td>");
out.println("<td>清除缓冲区的当前内容</td>");
out.println("<td> ");
out.println("</td>");
out.println("</tr>");

out.println("<tr>");
out.println("<td>close()</td>");
out.println("<td>关闭输出流</td>");
out.println("<td> </td>");
out.println("</tr>");

out.println("<tr>");
out.println("<td>flush()</td>");
out.println("<td>强制输出缓冲区的数据</td>");
out.println("<td> </td>");
out.println("</tr>");

out.println("<tr>");
out.println("<td>isAutoFlush()</td>");
out.println("<td>判断缓冲区是否具有强制输出 autoFlush 的功能</td>");
out.println("<td>");
out.isAutoFlush()
out.println("</td>");
out.println("</tr>");

out.println("<tr>");
out.println("<td>getBufferSize()</td>");
out.println("<td>返回缓冲区的大小</td>");
out.println("<td>");
out.getBufferSize();
out.println("</td>");
out.println("</tr>");

out.println("<tr>");
out.println("<td>getRemaining()</td>");
out.println("<td>返回缓冲区没有占用的空间</td>");
out.println("<td>");
out.getRemaining();
out.println("</td>");
out.println("</tr>");

out.println("<tr>");
out.println("<td>newLine()</td>");
out.println("<td>输出一新行</td>");
```

```
        out.println("<td>");
        newLine();
        out.println("</td>");
        out.println("</tr>");

        out.println("<tr>");
        out.println("<td>print()</td>");
        out.println("<td>输出信息</td>");
        out.println("<td>输出文字</td>");
        out.println("</tr>");

        out.println("<tr>");
        out.println("<td>println()</td>");
        out.println("<td>换行输出信息</td>");
        out.println("<td>换行输出一行</td>");
        out.println("</tr>");
    %>
    </table>
    </body>
    </html>
```

(1) 如果在 clear()方法一行中，增加 clear()方法的示例，问运行结果会发生怎样的变化？为什么发生这样变化，原因是什么？

(2) 如果为 clearBuffer()一行中，增加 clearBuffer()方法的示例，问运行结果又会发生什么变化，比较 clear()方法和 clearBuffer()方法的异同？

(3) 观察 out 对象的 print()和 println()方法在 JSP 页面运行结果有什么不同？

(4) 如果通过 page 指令默认 JSP 页面的缓冲区的大小是多少，用 page 指令分别设置缓冲区大小为 1kb、2kb 和 5kb，观察程序的运行结果会发生什么变化？

练习 2：输出 XHTML MP 菜单。

设计一个服务器端的 JSP 文件 menu.jsp，该文件可以动态生成 WML 菜单页面，该菜单页面中的菜单项有“新闻”、“财经”、“娱乐”、“游戏”、“论坛”以及动态生成当天的时间。具体要求是：补充 menu.jsp 对应的程序清单 9-2，用 out 对象输出具有链接的菜单项，并用跑马灯的形式显示访问该页面的时间。运行结果如图 9-2。



图 9-2　out 对象输出菜单

程序清单 9-2：

```
<!--menu.jsp-->
<%@page contentType="text/html; charset=gb2312" language="
java" import="java.util.Date" %>
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD XHTML Mobile 1.0//
EN"
"http://www.wapforum.org/DTD/xhtml-mobile10.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<meta http-equiv="refresh" content="60"/>
<title>WAP 网页菜单</title>
<style type="text/css">
    p{ display:-wap-marquee;
        -wap-marquee-loop:10;
        -wap-marquee-style:slide;
          }
</style>
</head>

<body>
<%
  out.println("新闻");
  out.println("财经");
  out.println("娱乐");
  out.println("游戏");
  out.println("论坛");
%>
<%
  out.println("<p>");
  out.println("登录时间是:");
  out.println("</p>");
%>
</body>
</html>
```

9.3　实验 9.2　内置对象的 4 种作用域

实验目的:

(1) 了解内置对象的 4 种作用域以及不同对象所具有的作用域。

(2) 比较 4 种内置对象作用域的不同之处。

(3) 初步了解 9 种内置对象的具体作用和常见方法。

(4) 初步运用 9 种内置对象解决实际应用问题。

实验内容:

(1) 观察不同内置对象的作用域下实现对表格背景颜色的设置。

(2) 利用 session 实现一个购物车程序,实现将虚拟商店的数码产品放置在购物车内。

(3) 利用 application 对象制作一个简易留言板。

实验步骤：

练习 1：设置表格的背景。

本次练习是初步了解内置对象的作用域，比较 4 种作用域的异同。并对常见的内置对象的常见方法具有一定的了解。按照下列的要求完成实验任务。

（1）阅读并分别运行下列程序 first.jsp 和 second.jsp，代码见程序清单 9-3 和程序清单 9-4 记录运行结果，观察并记录不同程序下输出 bgColor 属性的值，并分析运行结果。

程序清单 9-3：

```
<!--first.jsp-->
<%@page contentType="text/html; charset=gb2312" language="java" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>设置的背景 1</title>
</head>

<body bgcolor="#FFFFFF">
<%
  pageContext.setAttribute("bgColor","#FFFFFF");
  out.println("背景颜色：");
  out.println(pageContext.getAttribute("bgColor"));
%>
</body>
</html>
```

程序清单 9-4：

```
<!--second.jsp-->
<%@page contentType="text/html; charset=gb2312" language="java" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>设置背景</title>
</head>

<body bgcolor="#FFFFFF">
<%
  out.println("背景颜色：");
  out.println(pageContext.getAttribute("bgColor"));
%>
```

```
</body>
</html>
```

（2）阅读并分别运行下列程序 third.jsp 和 forth.jsp，代码见程序清单 9-5 和程序清单 9-6，记录运行结果，观察并记录不同程序下输出 bgColor 属性的值？然后在程序清单 9-5 的 ① 处去掉注释，并添加代码“<jsp:forward page="forth.jsp"/>”，分析运行结果，比较添加代码前后的异同。

程序清单 9-5：

```
<!--third.jsp-->
<%@page contentType="text/html; charset=gb2312" language="java" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>设置表格背景 3</title>
</head>

<body>
<%
  request.setAttribute("bgColor","#FF0000");
  String color=""+request.getAttribute("bgColor");
  out.println("背景颜色 1: ");
  out.println("<table border='1'>");
  out.println("<tr><td>颜色</td><td>值</td></tr>");
  out.println("<tr><td>"+color+"</td>");
  out.println("<td bgcolor='"+color+"'> </td>");
  out.println("</table>");
%>
<!--①-->
</body>
</html>
```

程序清单 9-6：

```
<!--forth.jsp-->
<%@page contentType="text/html; charset=gb2312" language="java" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>设置表格背景</title>
</head>
```

```
<body>
<%
  String color=""+request.getAttribute("bgColor");
  if(color==null)
     color="#FFFFFF";
  out.println("背景颜色 2: ");
  out.println("<table bgcolor="+color+">");
  out.println("<tr><td>颜色</td>");
  out.println("<tr><td>"+color+"</td>");
  out.println("</table>");
%>
</body>
</html>
```

(3) 将程序清单 9-5 和程序清单 9-6 代码中的 request 对象用 session 对象替代,然后在同一浏览器窗口中分别运行程序清单 9-5 和程序清单 9-6 的代码,观察运行结果。关闭浏览器窗口后再打开,再次重新运行修改后的程序清单 9-6,比较与前次运行结果的不同。如有不同,写出原因。

(4) 将程序清单 9-5 和程序清单 9-6 代码中的 request 对象用 application 对象替代,然后在同一浏览器窗口中分别运行程序清单 9-5 和程序清单 9-6 的代码,观察运行结果。关闭浏览器窗口后再打开,再次重新运行修改后的程序清单 9-6。比较用 application 和 session 对象替换的运行结果的不同之处,写出原因。

(5) 定义一个 web.xml,代码如程序清单 9-7 所示。将程序清单 9-8 的 fifth.jsp 的 代码 1 ～ 代码 3 处补充完整,实现用 config 对象来获取并设置表格的前景颜色和背景颜色,运行结果见图 9-3。修改 web.xml 的 bgcolor 参数的值,重新启动 Tomcat,观察分别用 fifth.jsp 和 tablecolors 名的运行结果。比较二者的不同之处。



图 9-3　设置表格颜色

程序清单 9-7:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!--web.xml-->

<web-app xmlns="http://java.sun.com/xml/ns/javaee"
```

```
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
    version="2.5">

    <display-name>An application of setting colors for a table</display-name>
    <description>
    An Web application for setting colors for a table.
    </description>

    <servlet>
     <servlet-name>TableColors</servlet-name>
     <jsp-file>/fifth.jsp</jsp-file>
     <init-param>
         <param-name>color</param-name>
         <param-value>#FFFFFF</param-value>
     </init-param>
     <init-param>
         <param-name>bgcolor</param-name>
         <param-value>#0000FF</param-value>
     </init-param>
    </servlet>

    <!--定义 Servlet 的映射 -->
    <servlet-mapping>
     <servlet-name>TableColors</servlet-name>
         <url-pattern>/tablecolors</url-pattern>
    </servlet-mapping>
</web-app>
```

程序清单 9-8：

```
<!--fifth.jsp -->
<%@page contentType="text/html; charset=gb2312" language="java" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>设置表格颜色</title>
</head>

<body>
<%
  String color= 代码 1 ;
//获取前景色
  String bgcolor= 代码 2 ;
```

```
//获取背景色
  out.println( 代码 3 );
//设置表格的前景色和背景色
  out.println("<tr>");
  out.println("<th>前景色</th>");
  out.println("<th>背景色</th>");
  out.println("</tr>");
  out.println("<tr>");
  out.println("<td>"+color+"</td>");                    //输出前景色的值
  out.println("<td>"+bgcolor+"</td>");                  //输出背景色的值
  out.println("</tr>");
  out.println("</table>");
%>
</body>
</html>
```

练习 2：用 session 对象实现购物车。

本次练习主要是深入了解会话管理以及 session 对象的具体应用。具体内容是将虚拟商店的数码产品放置在购物车内。产品的类别有闪存盘、移动硬盘、摄像头、数码相机和 MP3/MP4。每个类别有具体品牌的产品。要求，设计一个购物车，客户可以选择产品放置在购物车中，并可以浏览购物车内的产品。注意，客户购物车的产品可用 session 对象保存。运行结果类似图 9-4 至图 9-7。



图 9-4　产品类别运行结果

地址(D) http://jspcourse.edu.cn:8080/digital/product.jsp?id=2

顾客选择的产品:

移动硬盘
联想稳定型Ⅱ代(40G)
单价:899元
购买数量: 2
联想稳定型Ⅱ代(60G)
单价:119元
购买数量:
联想稳定型Ⅱ代(80G)
单价:1499元
购买数量:
购买 清除

图 9-5　某类别的产品运行结果

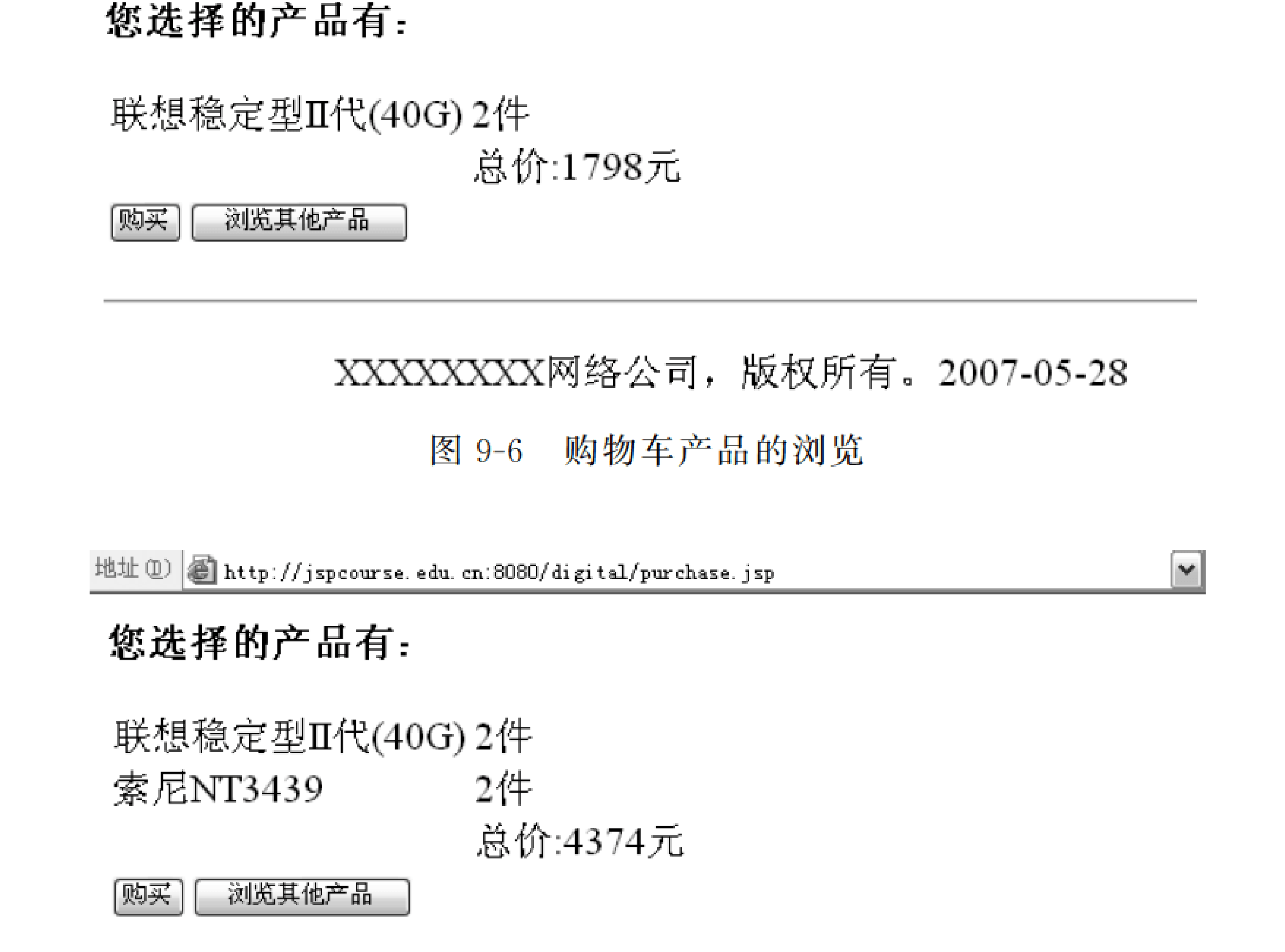


图 9-6 购物车产品的浏览

图 9-7 继续购物后购物车产品浏览

练习 3：用 application 对象实现一个简易的留言板。

设计一个简单的留言板。留言板提供姓名、标题和留言 3 个信息的输入。用户可以提交留言、浏览留言板的历史记录以及重置留言信息。确认留言后，发送的所有信息以表格形式在同一页显示出来，运行结果如图 9-8 所示。要求用 JSP 内置对象 application 来保存留言信息，不需要应用数据库。

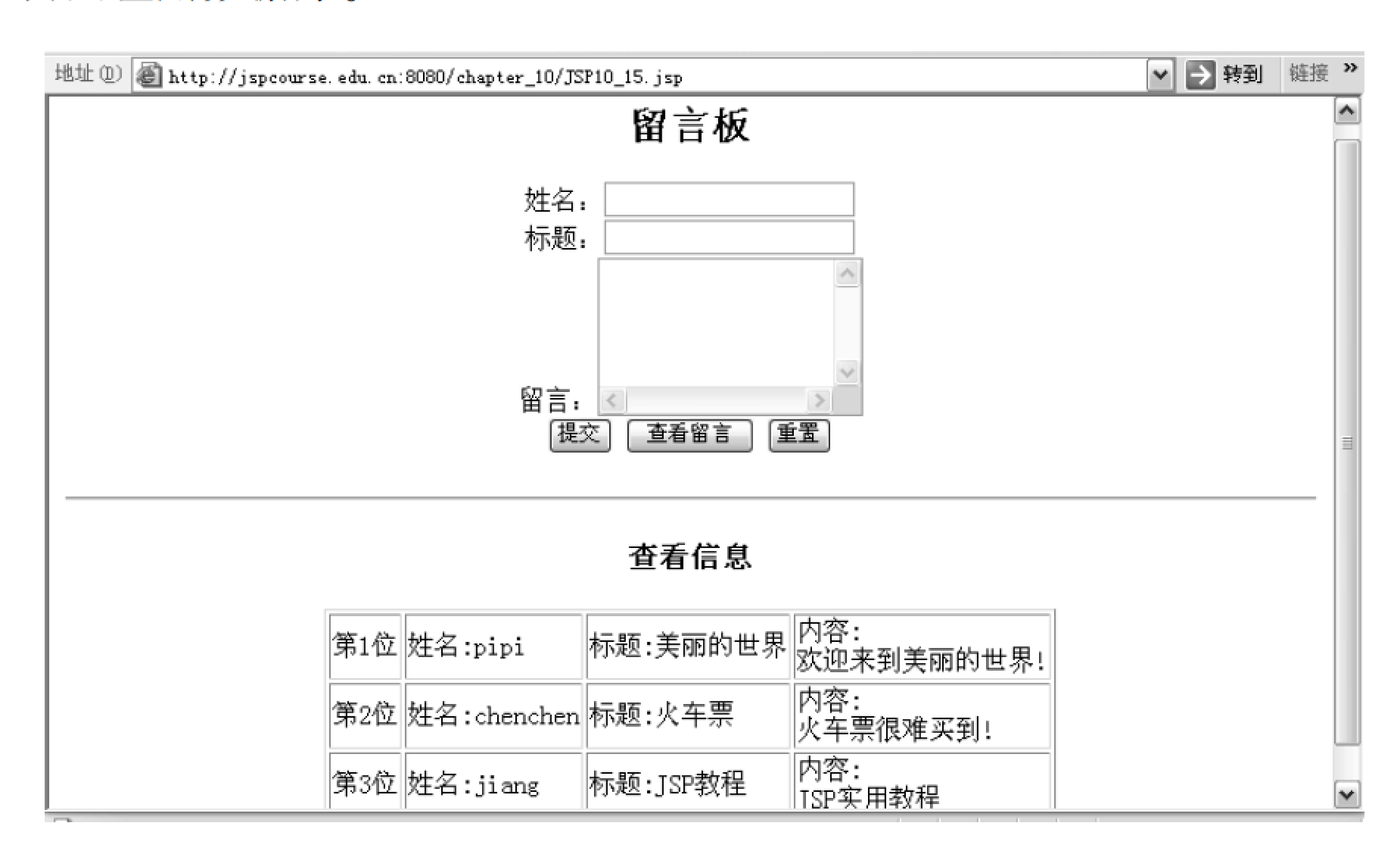


图 9-8 留言板的运行结果

(1) 比较 page、request、session 和 application 作用范围的不同之处。

(2) 要在不同页面之间共享数据,可以使用什么方法?要在不同用户之间共享数据,可以使用什么方法?

9.4 实验 9.3 创建错误处理页面

实验目的:

(1) 了解 exception 对象的常见方法以及用 exception 对象解决实际问题。

(2) 初步了解 JSP 错误处理的实现过程。

(3) 进一步体会内置对象的作用域的概念。

实验内容:

设计并实现网站的用户登录,如果用户的输入的关键信息如(用户名或密码为空),要求给出具体错误提示信息;如果用户输入的信息 3 次错误,则要求退出用户登录界面,并给出相应的错误原因。

实验步骤:

本次实验是了解当错误发生时,该如何处理错误。程序清单 9-9~9-11 分别是程序 log.jsp、validate.jsp 和 error.jsp 的部分代码。要求如下:

(1) 修改 log.jsp 实现登录界面,使登录失败的相关信息可以显示。

(2) 修改 validate.jsp 实现登录信息的验证。验证用户是否输入密码、是否输入用户名、是否输入的信息正确。如果用户输入的信息的次数超过三次,要求退出登录,返回主页面;如果用户输入的用户名为"chenyi"及密码为"123456",则输出登录成功的信息。

(3) 将 error.jsp 的 代码段 补充完整,实现对错误的处理。运行结果如图 9-9 至图 9-15 所示。

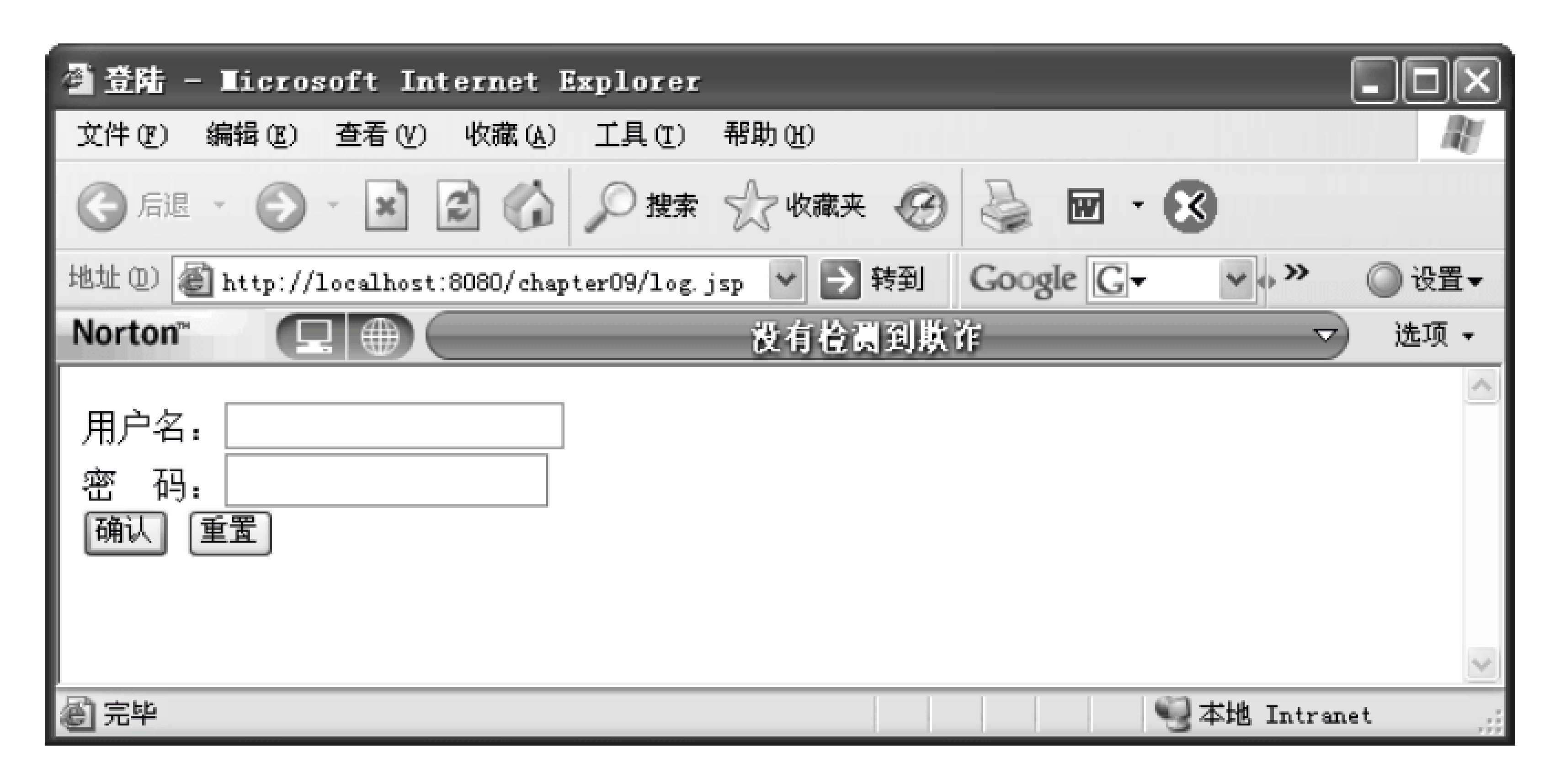


图 9-9 输入信息为空



图 9-10　提示输入信息为空的错误信息



图 9-11　输入信息错误

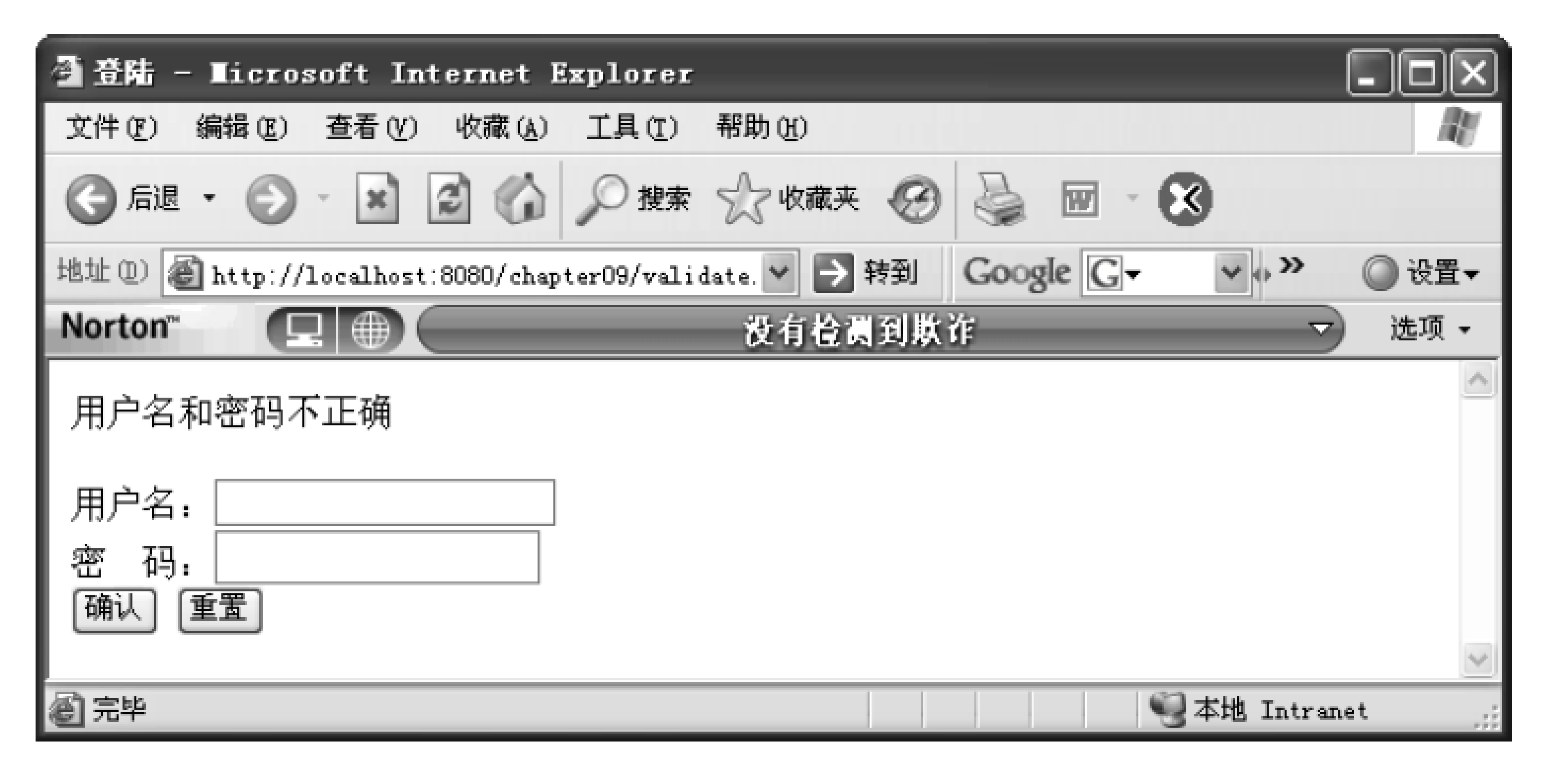


图 9-12　提示输入信息错误

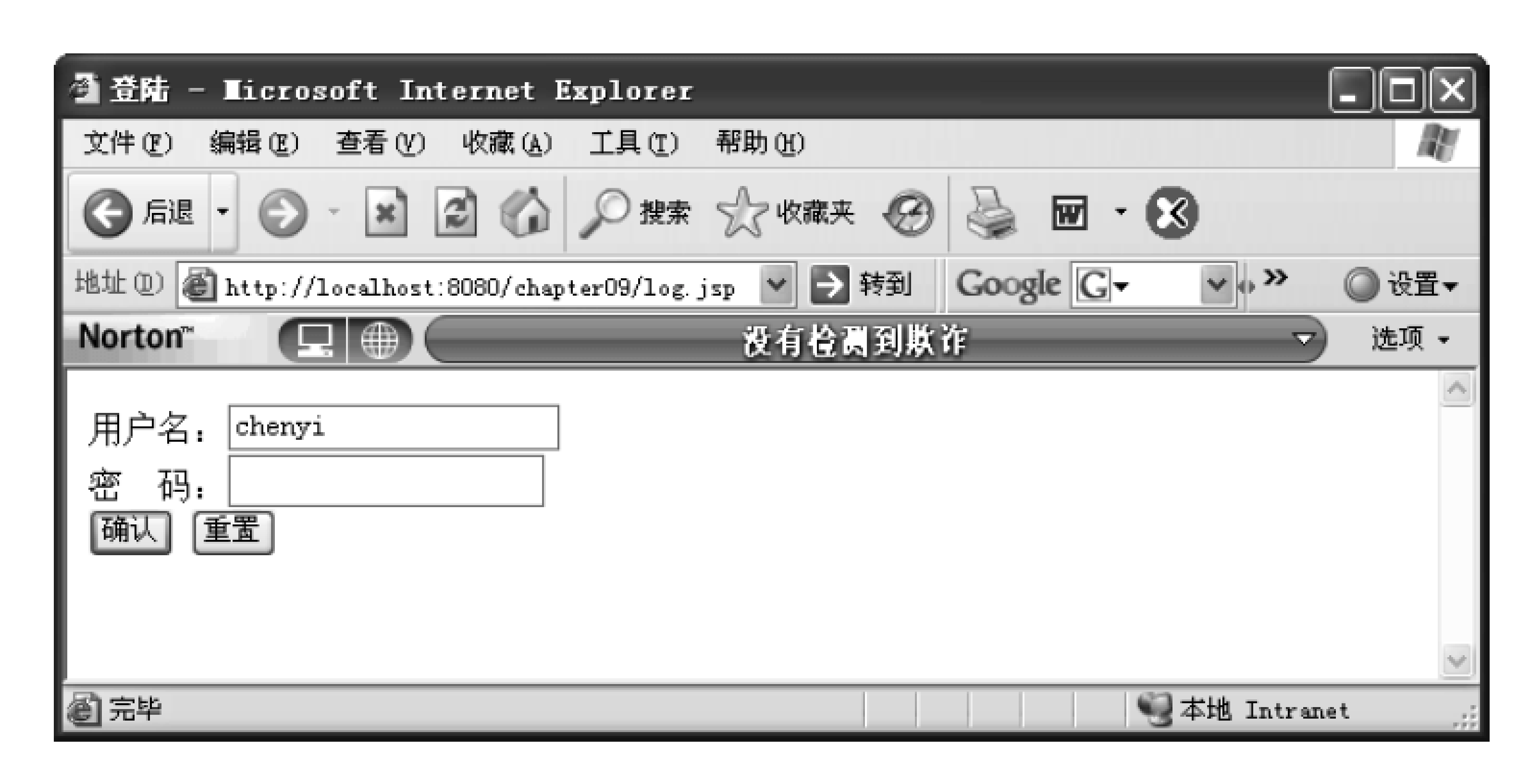


图 9-13　输入密码为空

图 9-14　显示输入密码为空的提示信息



图 9-15　输入三次登录信息错误后转向主页

程序清单 9-9：

```
<!--log.jsp-->
<%@page contentType="text/html; charset=gb2312" language="java"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>登录</title>
</head>

<body>
<form method="get" action="validate.jsp">
    用户名：
<input type="text" name="username" width="10"/>
<br/>
    密   码：
<input type="password" name="password" width="10"/>
<br/>
    <input type="submit" value="确认"/>
    <input type="reset" value="重置"/>
</form>
```

```
</body>
</html>
```

程序清单 9-10：

```
<!--validate.jsp-->
<%@page contentType="text/html; charset=gb2312" language="java"  errorPage="error.jsp"
%>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>验证</title>
</head>

<body>
<%
   String userStr=request.getParameter("username");
   String passStr=request.getParameter("password");

   if(userStr.equals("chenyi")&&passStr.equals("123456"))
    out.println("登录成功！"+times);
%>
</body>
</html>
```

程序清单 9-11：

```
<!--error.jsp-->
<%@page contentType="text/html;charset=gb2312"language="java"isErrorPage="true"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>警告！</title>
</head>

<body>
<%
   代码段
    //补充关于错误的处理；
%>
    <jsp:forward page="log.jsp"/>
</body>
</html>
```

第 10 章　JSP 的文件操作

文件操作在 JSP 中也占据着非常重要的地位。开发人员经常会涉及对文件、文件目录的读写，由于 JSP 是基于网络的开发，很多程序都涉及到文件的上传和下载，这也是属于 JSP 对文件的操作，这些对文件的操作都是使用了 JSP 的输入和输出流来完成的。本章将介绍 JSP 文件操作的实验，帮助读者掌握 JSP 的文件操作。

10.1　预 备 知 识

在 Java 语言的程序设计中，I/O 操作，特别是文件操作是非常重要的一部分，同样，文件操作在 JSP 中也占据着非常重要的地位。由于 JSP 能使用 Java 的 I/O 系统，所以在 JSP 中对文件的操作和一般 Java 程序没有太大的区别。本章本着从浅及深的角度，首先介绍 java. io. File 类，然后介绍 JSP 中常用的输入输出流，最后对一个常用于文件上传处理的 jspSmartUpload 组件展开介绍。

10. 1. 1　File 类

java. io. File 类专门提供一种抽象，用于以平台独立的方式处理大多数平台依赖的、复杂的文件和路径名问题。File 类包含许多获取文件属性的方法以及创建、删除和重命名文件和文件目录的方法。

1. File 的创建

File 的主要构造方法如下：

(1) File(String pathname)；

(2) File(String parent, String child)；

(3) File(File parent,String child)。

2. File 的主要方法

File 的主要方法见表 10-1。

表 10-1　File 的主要方法

方　　法	功　　能
boolean canRead()	测试应用程序是否能从指定的文件中进行读取
boolean canWrite()	测试应用程序是否能写当前文件
boolean delete()	删除当前对象指定的文件
boolean exists()	测试当前 File 是否存在
String getAbsolutePath()	返回由该对象表示的文件的绝对路径名
String getCanonicalPath()	返回当前 File 对象的路径名的规范格式
String getName()	返回表示当前对象的文件名
String getParent()	返回当前 File 对象路径名的父路径名，如果此名没有父路径则为 null

方　　法	功　　能
String getPath()	返回表示当前对象的路径名
boolean isAbsolute()	测试当前 File 对象表示的文件是否是一个绝对路径名
boolean isDirectory()	测试当前 File 对象表示的文件是否是一个路径
boolean isFile()	测试当前 File 对象表示的文件是否是一个普通文件
long lastModified()	返回当前 File 对象表示的文件最后修改的时间
long length()	返回当前 File 对象表示的文件长度
String[]list()	返回当前 File 对象指定的路径文件列表
File[] listFiles()	返回一个抽象路径名数组,这些路径名表示此抽象路径名所表示目录中的文件
boolean mkdir()	创建一个目录,它的路径名由当前 File 对象指定
boolean renameTo(File)	将当前 File 对象指定的文件更名为给定参数 File 指定的路径名
String toString()	返回当前对象的字符串表示

10.1.2　JSP 的输入流和输出流

1. JSP 的输入和输出流

File 对象封装了文件或路径属性,但是不包含从文件中读写数据的方法。为了进行文件读写操作,需要用适当的 Java I/O 类创建对象,这些对象包含从文件中读写数据的方法。Java 有许多用于各种目的的 I/O 类,可以把它们分为输入类和输出类。输入类包含读数据的方法,输出类包含写数据的方法。在 Java 中是使用建立 I/O 对象的方式进行输入和输出,输入的对象被称为输入流,输出的对象被称为输出流。

流是一个很形象的概念,当程序需要读取数据时,就会开启一个通向数据源的流,这个数据源可以是文件、内存或网络连接,则这就是“输入流”,如图 10-1 所示。类似的,当程序需要写入数据的时候,就会开启一个通向目的地的流,这就是“输出流”,如图 10-2 所示,这时你就可以想像数据好像在这其中“流”动一样。

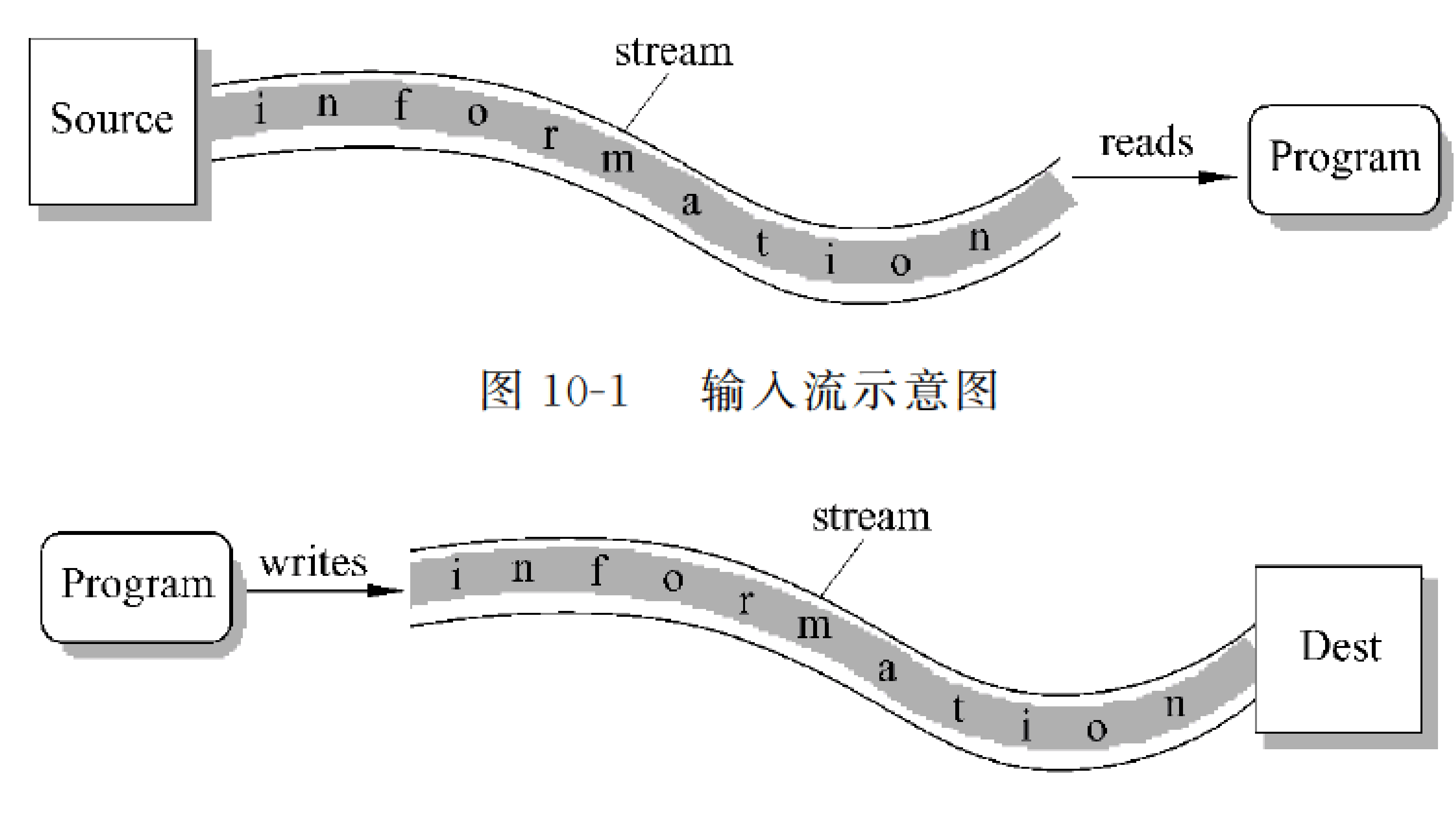


图 10-1　输入流示意图

图 10-2　输出流示意图

Java I/O 中的流分为两种,一种是字节流,另一种是字符流,分别由 4 个抽象类来表示:InputStream、OutputStream、Reader 和 Writer。Java 中其他多种多样变化的流均是由它们继承出来的。

2．字节流

以字节为单位进行读写的称为字节流，在 Java 中，字节流分为输入流和输出流，分别是 InputStream 和 OutputStream 及其子类。字节流结构如图 10-3 和图 10-4 所示。

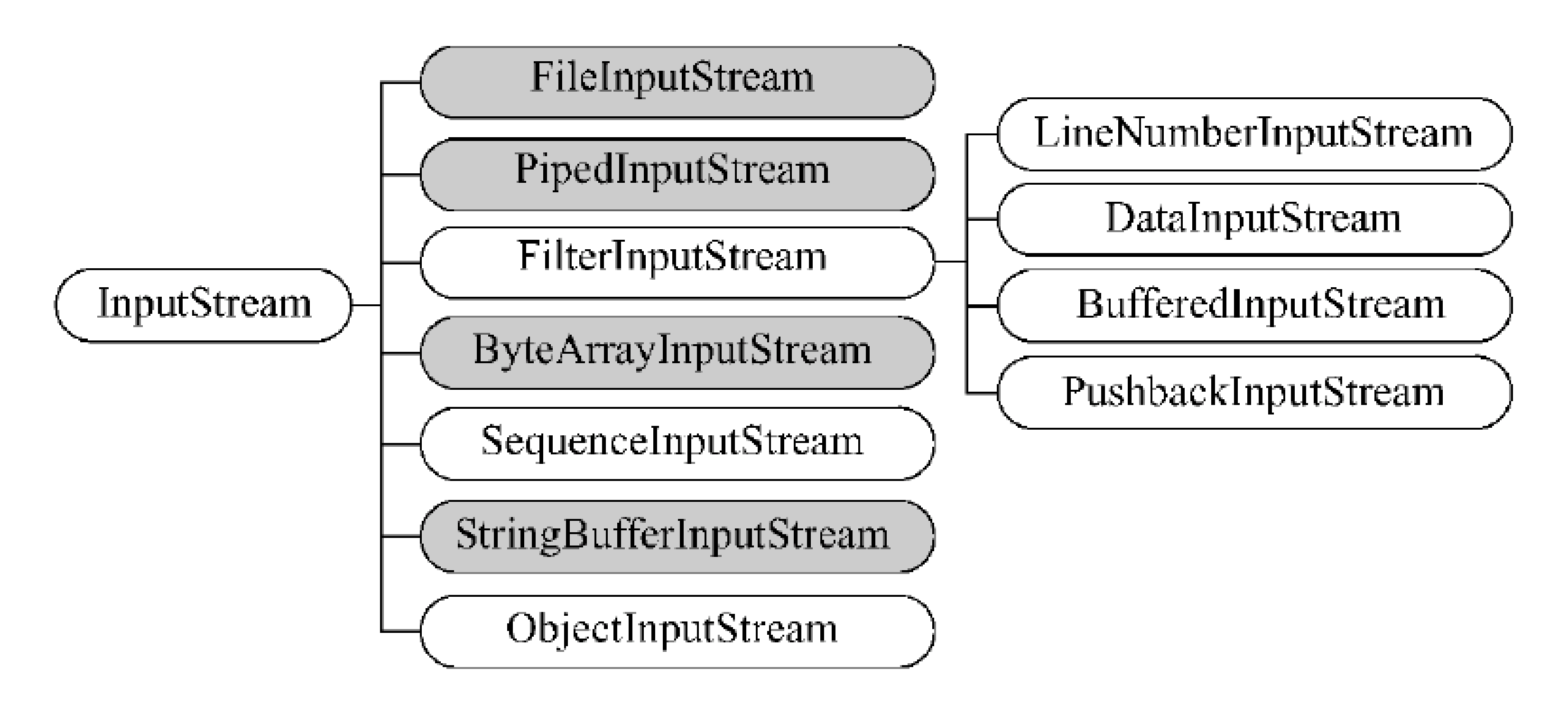


图 10-3　InputStream 类结构图

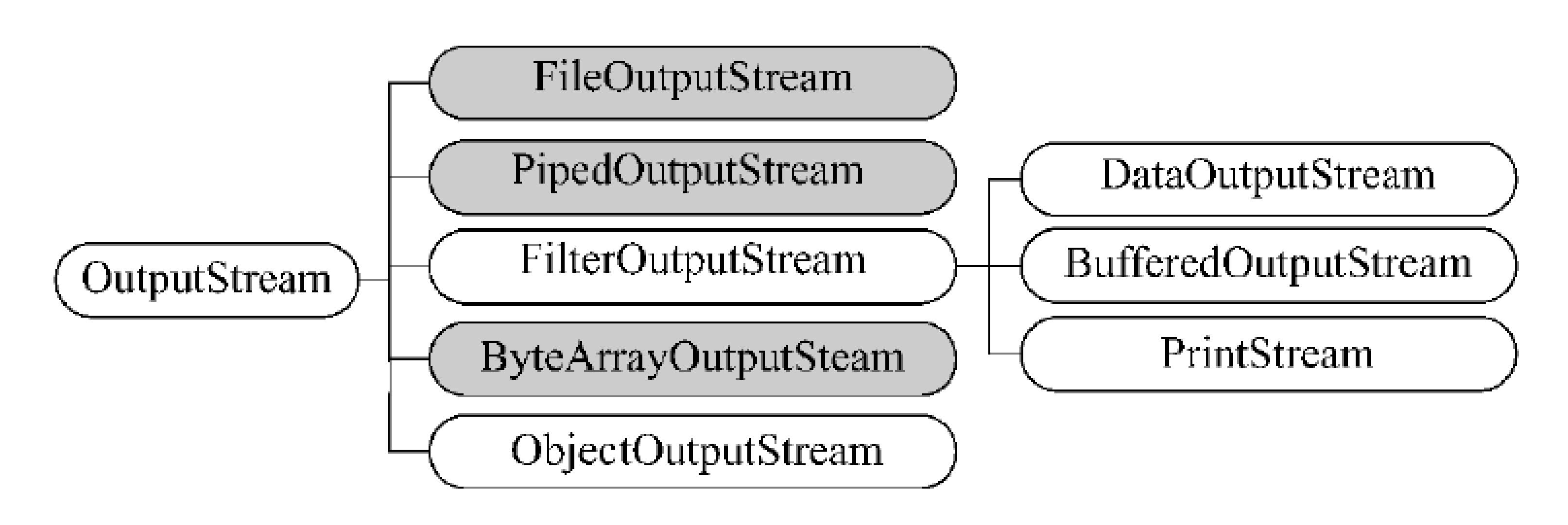


图 10-4　OutputStream 类结构图

表 10-2 是字节流的几个常用子类及功能。

表 10-2　字节流常用子类及功能

字 节 流 类	功能简单介绍
DataInputStream	包含了读取 Java 标准数据类型的输入流
DataOutputStream	包含了写 Java 标准数据类型的输出流
ByteArrayInputStream	从字节数组读取的输入流
ByteArrayOutputStream	写入字节数组的输出流
FileInputStream	从文件读入的输入流
FileOutputStream	写入文件的输出流
PrintStream	包含最常见的 print()和 println()方法实现输出流
PushbackInputStream	返回一个字节到输入流，主要用于编译器的实现
PipedInputStream	输入管道
PipedOutputStream	输出管道
SequenceInputStream	将 n 个输入流联合起来，一个接一个按一定顺序读取
BufferedInputStream	缓冲输入流
BufferedOutputStream	缓冲输出流
FilterInputStream	实现了 InputStream 接口的过滤器输入流
FilterOutputStream	实现了 OutputStream 接口的过滤器输出流

字节流抽象类 InputStream 和 OutputStream 的常用方法如表 10-3 所示。

表 10-3　InputStream 和 OutputStream 的常用方法

方　　法	功　　能
int read()	读取一个字节的数据，并返回读到的字节
int read(byte[]b)	将数据读入一个字节数组，同时返回读回的字节数
int read(byte[] b,int off,int len)	将数据读入一个字节数组。返回读回的实际字节数。其中，b：指定要把字节读入哪个数组；off：指定数组的偏移位置，第一个字节应放在哪个位置；len：读回的最大字节数
long skip(long *n*)	在输入流中跳过 *n* 个字节，它返回的实际跳过的字节数
int available()	返回在不加阻止的情况下，可用的字节数
void mark(int readlimit)	在此输入流中标记当前的位置
void reset()	将此流重新定位到对此输入流最后调用 mark 方法时的位置
long skip(long *n*)	跳过和放弃此输入流中的 *n* 个数据字节
void write(int b)	写入一个字节的数据
void write(byte[] b)	写入数组 b 内的所有字节
void write(byte[] b,int off,int len)	写入数组 b 内的所有字节，其中，b：指定要从哪个数组写入数据；off：指定数组 b 的一个偏移位置，从哪个位置的字节开始写入；len：要写入的字节数量
void flush()	清空输出流
void close()	关闭输入流或输出流

3. 字符流

Java 中的字节流是用于处理字节的输入和输出的，包括读写二进制数据等方面的内容。同字节流类似，字符流也是通过两个顶层的抽象类 Reader 和 Writer 的子类来实现对 Unicode 字符流的处理，如图 10-5 和图 10-6 所示。

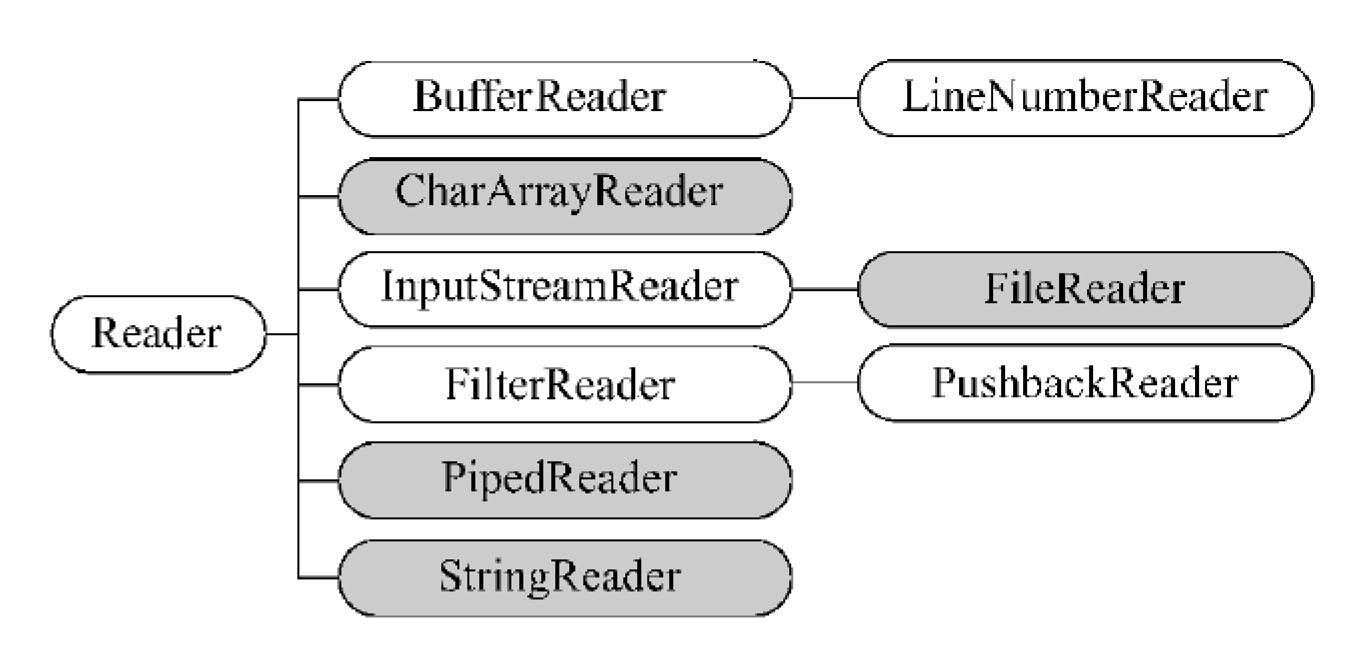


图 10-5　Reader 类结构图

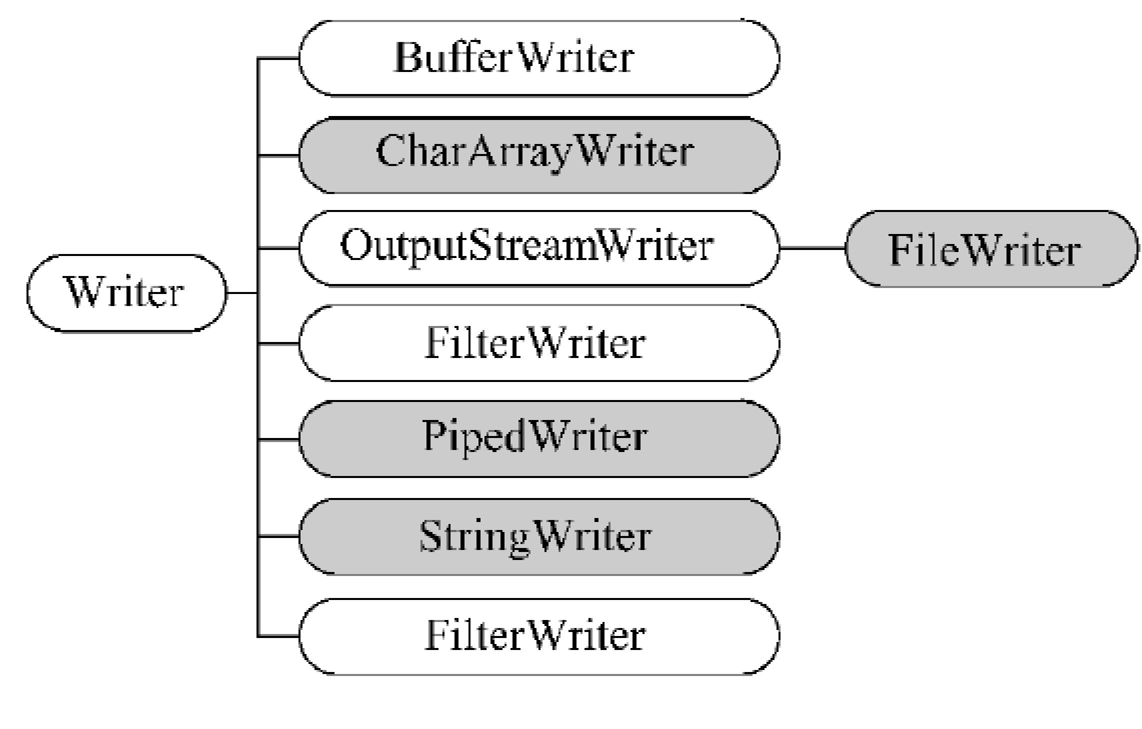


图 10-6　Writer 类结构图

表 10-4 是字符流的几个常用子类及功能。

表 10-4　字符流常用子类及功能

字 符 流 类	功能简单介绍
StringReader	从字符串读取的输入流
StringWriter	写入字符串的输出流
FileReader	从文件读入的输入流
FileWriter	写入文件的输出流
PushbackReader	返回一个字符到输入流，主要用于编译器的实现
PipedReader	输入管道
PipedWriter	输出管道
CharArrayReader	从字符数组读取的输入流
CharArrayWriter	写入字符数组的输出流
BufferReader	缓冲输入流
BufferWriter	缓冲输出流
FilterReader	实现了 InputStream 接口的过滤输入流
FilterWriter	实现了 OutputStream 接口的过滤输出流
InputStreamReader	将字节转换为字符的输入流
OutputStreamWriter	将字节转换为字符的输出流

10.1.3　文件上传

文件上传功能使用免费的上传组件 jspSmartUpload 来实现，jspSmartUpload 组件可以免费下载，下载后文件的名字是 jspSmartUpload.zip。解压该文件后只需要将 web-inf 目录下的文件复制到 Web 应用程序所在目录的 WEB-INF 文件夹下就可以在 Web 程序中使用上传组件了。

jspSmartUpload 包含有 5 个类。

(1) File：上传文件的抽象表示类。

(2) Files：包含多个上传文件的 File 实例。

(3) Request：等价于 Servlet 的 ServletRequest 类。

(4) SmartUpload：实现上传的类。

(5) SmartUploadException：上传抛出的异常类。

10.2　实验 10.1　JSP 中文件读写

实验目的：

(1) 掌握 JSP 中读写文件的方法。

(2) 了解 File 类的创建和常用的方法。

(3) 了解随机读写文件类 RandomAccessFile 的运行原理和常用的方法。

(4) 练习用 RandomAccessFile 实现动态文件的读写。

实验内容：

本次实验用于读者掌握JSP中对文件的读写技术，实验完成了一个基于Web联系人地址管理程序，实现了联系人地址的添加、删除、修改，其中联系人地址信息的保存采用服务器端的文本文件，利用RandomAccessFile来实现读写。本次实验由3个练习构成。

(1) 为RandomAccessFile读写文件设计后台JavaBean程序。

(2) 实现联系人地址的读取。

(3) 实现联系人地址的更新操作，包括新建、修改、删除。

实验步骤：

练习1：设计后台Java程序处理文件读写。

(1) 启动Java开发环境，新建一个Java文件FixedLengthStringIO.java，用来实现固定长度的字符串的读写，当字符串长度不足固定长度，用空格来补足；如果字符串超过固定长度则截断。将程序清单10-1代码输入。

程序清单10-1：

```
//FixedLengthStringIO.java
import java.io.*;

public class FixedLengthStringIO {
    //从输入流 in 中读入固定长度的字符串
    public static String readFixedLengthString(int size, DataInput in)
            throws IOException {
        //定义字符数组
        char[]chars=new char[size];
        //读取 size 个字符
        for (int i=0; i<size; i++)
            chars[i]=in.readChar();
        return new String(chars);
    }

    //向输出流 out 写入固定长度的字符串
    public static void writeFixedLengthString(String s, int size, DataOutput out)
            throws IOException {
        char[] chars=new char[size];
        //将字符串 s 转换成字符数组 chars
        s.getChars(0, s.length(), chars, 0);
        //将字符串截断或者补充空格来控制字符串为固定长度,
        for (int i=Math.min(s.length(), size); i<chars.length; i++)
            chars[i]=' ';
        //写入字符串
        out.writeChars(new String(chars));
    }
}
```

（2）编译程序清单 10-1 中的 Java 代码 FixedLengthStringIO. java，检查该代码是否能够实现固定长度字符串的读写。

（3）新建 Java 文件 AddressBookIO. java，利用程序清单 10-1 中的代码和随机读写文件类 RandomAccessFile 实现对文本地址簿的读写，将程序清单 10-2 中代码输入。

程序清单 10-2：

```
//AddressBookIO.java
import java.io.*;

public class AddressBookIO {
    //定义各个字段的固定长度
    final static int NAME_SIZE=32;                         //名字
    final static int STREET_SIZE=32;                       //地址
    final static int CITY_SIZE=20;                         //城市
    final static int STATE_SIZE=10;                        //省
    final static int ZIP_SIZE=6;                           //邮编
    final static int RECORD_SIZE=                          //一条记录的长度
    (NAME_SIZE+STREET_SIZE+CITY_SIZE+STATE_SIZE+ZIP_SIZE);

    private RandomAccessFile raf;                          //指定关联文件
    private int no;
    private String name="";
    private String street="";
    private String city="";
    private String state="";
    private String zip="";

    public AddressBookIO(RandomAccessFile initRaf) {
        raf=initRaf;
        no=0;
    }

    //get 和 set 方法
    public int getAddressCount() {
        int count=0;
        try {
            long lastPosition=raf.length();
            count=(int) lastPosition/(2*RECORD_SIZE);
        } catch (Exception e) {
        }
        return count;
    }

    public String getName() {
```

```
        return name;
    }

    public void setName(String name) {
        this.name=name;
    }

    public String getStreet() {
        return street;
    }

    public void setStreet(String street) {
        this.street=street;
    }

    public String getCity() {
        return city;
    }

    public void setCity(String city) {
        this.city=city;
    }

    public String getState() {
        return state;
    }

    public void setState(String state) {
        this.state=state;
    }

    public String getZip() {
        return zip;
    }

    public void setZip(String zip) {
        this.zip=zip;
    }

    //写入文件方法,position 为写入位置
    public void writeAddress(long position) throws IOException {
        raf.seek(position * 2 * RECORD_SIZE);
        FixedLengthStringIO.writeFixedLengthString(name, NAME_SIZE, raf);
        FixedLengthStringIO.writeFixedLengthString(street, STREET_SIZE, raf);
```

```
        FixedLengthStringIO.writeFixedLengthString(city, CITY_SIZE, raf);
        FixedLengthStringIO.writeFixedLengthString(state, STATE_SIZE, raf);
        FixedLengthStringIO.writeFixedLengthString(zip, ZIP_SIZE, raf);
    }

    //从文件中读出内容
    public void readAddress(long position) throws IOException {
        raf.seek(position * 2 * RECORD_SIZE);
        name=FixedLengthStringIO.readFixedLengthString(NAME_SIZE, raf);
        street=FixedLengthStringIO.readFixedLengthString(STREET_SIZE, raf);
        city=FixedLengthStringIO.readFixedLengthString(CITY_SIZE, raf);
        state=FixedLengthStringIO.readFixedLengthString(STATE_SIZE, raf);
        zip=FixedLengthStringIO.readFixedLengthString(ZIP_SIZE, raf);
    }

    public void addAnAddress()throws IOException{
        writeAddress(getAddressCount());
    }

    public void editAnAddress(long position)throws IOException{
        writeAddress(position);
    }

    public void delAnAddress(long position)throws IOException{
        for(long i=position;i<getAddressCount()-1;i++){
            readAddress(i+1);
            writeAddress(i);
        }
        raf.setLength(raf.length()-2 * RECORD_SIZE);
    }
}
```

（4）编译程序清单 10-2 中的 Java 代码 AddressBookIO.java，检查该代码是否能够实现对文本文件的随机读写，并注意该代码实现文本地址簿读写所需要的参数。

练习 2：实现联系人的读取。

打开 JSP 开发环境，新建一个 JSP 文件 addresslistR.jsp，部分代码如程序清单 10-3 所示，该 JSP 程序用于实现对地址文本文件 address.txt 的读取。请将程序清单的 代码 1 ～ 代码 3 补充完整，在 Tomcat 服务器中运行程序，使之能将 address.txt 中的地址信息显示，运行结果如图 10-7 所示。

程序清单 10-3：

```
<!--addresslistR.jsp-->
<%@page language="java" contentType="text/html; charset=gb2312"%>
<%@page import="java.io.*,chapter10.*"%>
```

```
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=gb2312">
    <title>地址簿列表</title>
  </head>

  <body>
  <%
      String path=getServletContext().getRealPath(".")+File.separator
              +"ch10"+File.separator+"address.txt";
      RandomAccessFile raf= 代码 1 ;
      //读取文件地址簿列表
      AddressBookIO addressIO=new AddressBookIO(raf);
        String delPosition=request.getParameter("position");
        //获取 position 参数的值
        if(delPosition!=null){
          addressIO.delAnAddress(Long.parseLong(delPosition));
    }
      out.print("<h1>地址簿列表</h1>");
      out.print("<table border=1 cellspacing=1>");
      out.print("<tr><td width=\"20%\">姓名</td>");
      out.print("<td width=\"40%\">地址</td>");
      out.print("<td width=\"10%\">城市</td>");
      out.print("<td width=\"10%\">省</td>");
      out.print("<td width=\"20%\">邮编</td></tr>");
      try {
        for (int i=0; i< 代码 2 ; i++) {
          //依次显示所有的地址信息
          addressIO. 代码 3 ;
          //读取第 i 条地址信息
            out.print("<tr><td>"+addressIO.getName()+"</td>");
            out.print("<td>"+addressIO.getStreet()+"</td>");
            out.print("<td>"+addressIO.getCity()+"</td>");
            out.print("<td>"+addressIO.getState()+"</td>");
            out.print("<td>"+addressIO.getZip()+"</td></tr>");
        }
      }catch (Exception e) {}
      out.print("</table>");
  %>
  </body>

</html>
```



图 10-7　地址簿读取列表

练习 3：实现联系人地址的更新操作，包括新建、修改和删除操作。

（1）修改程序清单 10-3 中 addresslistR.jsp，并把它保存为 addresslist.jsp，在其中添加联系人地址的新建、修改、删除功能。

（2）仔细阅读下列程序清单 10-4 的 addresslistR.jsp，填充程序完整。其中，代码 1 加入新建联系人链接"新建"；代码 2 加入修改链接"编辑"；代码 3 加入删除链接"删除"。注意：对于修改和删除两个链接是利用编号来定位。然后启动 Tomcat，运行 addresslist.jsp，使得运行结果如图 10-8 所示。



图 10-8　加入"编辑"、"删除"链接后的地址簿读取列表

程序清单 10-4：

```
<!--addresslist.jsp -->
<%@page language="java" contentType="text/html; charset=gb2312"%>
<%@page import="java.io.*,chapter10.*"%>
```

```
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=gb2312">
    <title>地址簿列表</title>
  </head>
  <body>
  <%
      String path=getServletContext().getRealPath(".")+File.separator
                +"ch10"+File.separator+"address.txt";
      RandomAccessFile raf=new RandomAccessFile(path, "rw");
      //读取文件地址簿列表
      AddressBookIO addressIO=new AddressBookIO(raf);
      String delPosition=request.getParameter("position");
      if(delPosition!=null){
        addressIO.delAnAddress(Long.parseLong(delPosition));
      }
      out.print("<h1>地址簿列表</h1>");
      out.print(代码 1);
      //输出新建链接
      out.print("<table border=1 cellspacing=1>");
      out.print("<tr><td width=\"10%\">编辑</td>");
      out.print("<td width=\"10%\">删除</td>");
      out.print("<td width=\"20%\">姓名</td>");
      out.print("<td width=\"30%\">地址</td>");
      out.print("<td width=\"10%\">城市</td>");
      out.print("<td width=\"10%\">省</td>");
      out.print("<td width=\"10%\">邮编</td></tr>");
      try {
        for (int i=0; i<addressIO.getAddressCount(); i++) {
         addressIO.readAddress(i);
          out.println(代码 2);
          //输出编辑链接
          out.println(代码 3);
          //输出删除链接
          out.print("<td>"+addressIO.getName()+"</td>");
          out.print("<td>"+addressIO.getStreet()+"</td>");
          out.print("<td>"+addressIO.getCity()+"</td>");
          out.print("<td>"+addressIO.getState()+"</td>");
          out.print("<td>"+addressIO.getZip()+"</td></tr>");
        }
      } catch (Exception e) {
      }
      out.print("</table>");
  %>
```

```
  </body>
</html>
```

(3) 下列程序清单 10-5 定义了一个 JSP 文件 addressdetail.jsp，它用于实现地址的编辑和新建，在 addressdetail.jsp 中利用表单实现了单个地址信息的新建和输入，由于编辑和新建共用这个文件，所以依靠 mode 输入参数来区分这两种操作，如果 mode 的值为 new，表示新建地址，否则表示为编辑地址。请将 代码 4 ～ 代码 6 补充完整，实现单个地址信息的新建和输入功能。

程序清单 10-5：

```
<!--addressdetail.jsp-->
<%@page contentType="text/html; charset=gb2312" language="java"%>
<%@page import="chapter10.*,java.io.*"%>
<html>
  <head>
    <title>AddressBook</title>
  </head>
  <%
      String path=getServletContext().getRealPath(".")+File.separator
        +"ch10"+File.separator+"address.txt";
  RandomAccessFile raf=new RandomAccessFile(path, "rw");
  //读取文件地址簿列表
  AddressBookIO addressIO= 代码 4 ;
  //创建保存地址信息的对象 addressIO
  String action=request. 代码 5 ;
  //获取 mode 的值
    String actiontype="";
    String actionaim="";
    String editposition="";
    if ( 代码 6 ) {
      //判断 mode 参数的值是否是 new
      actiontype="新建一个地址信息";
      actionaim="addanaddress.jsp";
    } else {
      actiontype="编辑地址信息";
      editposition=request.getParameter("position");
      actionaim="editanaddress.jsp?position="+editposition;
      addressIO.readAddress(Long.parseLong(editposition));
      }
  %>
  <body>
    <h1><%=actiontype%></h1>
    <hr>
    <form action="<%=actionaim%>" method="post">
```

```
<table border="0" cellspacing="0">
  <tr>
    <td>姓名:</td>
    <td>
      <%
        if (action.equals("new"))
        out.print("<input type=\"text\" size=\"40\" name=\"name\">");
        else
        out.print("<input type=\"text\" size=\"40\" name=\"name\" value="
              +addressIO.getName()+">");
      %>
    </td>
  </tr>
  <tr>
    <td>地址</td>
    <td>
      <%
        if (action.equals("new"))
        out.print("<input type=\"text\" size=\"40\" name=\"street\">");
        else
          out.print("<input type=\"text\" size=\"40\" name=\"street\" value=\""
              +addressIO.getStreet()+"\">");
      %>
    </td>
  </tr>
  <tr>
    <td>城市:</td>
    <td>
      <%
        if(action.equals("new"))
        out.print("<input type=\"text\" size=\"40\" name=\"city\">");
        else
        out.print("<input type=\"text\" size=\"40\" name=\"city\" value=\""
                    +addressIO.getCity()+"\">");
      %>
    </td>
  </tr>
  <tr>
    <td>省:</td>
    <td>
      <%
        if (action.equals("new"))
          out.print("<input type=\"text\" size=\"40\" name=\"state\">");
        else
          out.print("<input type=\"text\" size=\"40\" name=\"state\" value=\""
```

```
                    +addressIO.getState()+"\">");
            %>
          </td>
        </tr>
        <tr>
          <td>邮编：</td>
          <td>
            <%
              if (action.equals("new"))
                out.print("<input type=\"text\" size=\"40\" name=\"zip\">");
              else
                out.print("<input type=\"text\" size=\"40\" name=\"zip\" value=\""
                    +addressIO.getZip()+"\">");
            %>
          </td>
        </tr>
      </table>
      <input type="submit" value="提交">
    </form>
  </body>
</html>
```

（4）在 Tomcat 服务器中测试列表文件 addressdetail.jsp，检查运行效果。

（5）新建 addanaddress.jsp 和 editanaddress.jsp，它们分别用于实现对一个地址信息的加入和修改，它们的部分代码如程序清单 10-6 和程序清单 10-7 所示。这两个 JSP 文件又需要用到练习 1 中设计的类 AddressBookIO 来实现对后台文件的读写。请仔细阅读这两个程序，将 代码 7 ～ 代码 10 补充完整，实现题意的要求。

程序清单 10-6：

```
<!--addanaddress.jsp -->
<%@page contentType="text/html; charset=gb2312" language="java"%>
<%@page import="chapter10.*,java.io.*"%>
<%!String trans(String origin) {
    String result=null;
    try {
      byte[] temp=origin.getBytes("iso-8859-1");
      result=new String(temp, "gb2312");
    } catch (UnsupportedEncodingException usee) {
    }
    return result;
  }
%>
<%
    String path=getServletContext().getRealPath(".")+File.separator
        +"ch10"+File.separator+"address.txt";
```

```
    RandomAccessFile raf=new RandomAccessFile(path, "rw");
    AddressBookIO addressIO=new AddressBookIO(raf);
    addressIO.setName(trans(request.getParameter("name")));
    addressIO.setStreet(trans(request.getParameter("street")));
    addressIO.setCity(trans(request.getParameter("city")));
    addressIO.setState(trans(request.getParameter("state")));
    addressIO.setZip(trans(request.getParameter("zip")));
    addressIO.代码 7;
    //添加新地址信息
    response.代码 8;
    //转向至 addresslist.jsp
%>
```

程序清单 10-7：

```
<!--editanaddress.jsp-->
<%@page contentType="text/html; charset=gb2312" language="java"%>
<%@page import="chapter10.*,java.io.*"%>
  <%!String trans(String origin) {
    String result=null;
    try {
      byte[] temp=origin.getBytes("iso-8859-1");
      result=new String(temp, "gb2312");
    } catch (UnsupportedEncodingException usee) {
    }
    return result;
  }
%>
<%
    String path=getServletContext().getRealPath(".")+File.separator
        +"ch10"+File.separator+"address.txt";
    RandomAccessFile raf=new RandomAccessFile(path, "rw");
    //读取文件地址簿列表
    AddressBookIO addressIO=new AddressBookIO(raf);
    String editPosition=request.getParameter("position");
    addressIO.setName(trans(request.getParameter("name")));
    addressIO.setStreet(trans(request.getParameter("street")));
    addressIO.setCity(trans(request.getParameter("city")));
    addressIO.setState(trans(request.getParameter("state")));
    addressIO.setZip(trans(request.getParameter("zip")));
    if(editPosition!=null)
      addressIO.代码 9;
      //修改地址信息
    response.代码 10;
      //跳转到 addresslist.jsp
%>
```

（6）参考程序清单 10-6 和 10-7，设计 delanaddress.jsp 文件，实现地址的删除。

（7）保存以上代码，并启动 Tomcat 服务器进行测试，检查程序是否能正常完成上述的要求，运行效果如图 10-9 所示。



图 10-9　对单个联系人地址进行修改

10.3　实验 10.2　JSP 中文件目录的访问

实验目的：

（1）掌握 JSP 中读写文件的方法。

（2）了解 File 类的创建和对文件信息的读取。

（3）了解如何判断目录和文件，并掌握多层次目录读写的方法。

实验内容：

本次实验用于读者掌握 JSP 中对文件和目录信息的读写技术，实验完成了服务器当前地址下文件和目录列表的访问的任务。本次实验由两个练习构成。

（1）读取当前目录下文件并用列表显示。

（2）读取当前目录下文件和目录列表，并实现目录列表的层次访问。

实验步骤：

练习 1：读取当前目录下文件并用列表显示。

（1）启动 JSP 开发环境，新建一个 JSP 文件 simplefilelist.jsp，用来实现对服务器目录的文件访问。将程序清单 10-8 代码输入。

程序清单 10-8：

```
<!--simplefilelist.jsp -->
<%@page contentType="text/html; charset=gb2312" language="java"%>
```

```
<%@page import="java.io.*"%>
<html>
  <head>
    <title>简单文件列表</title>
  </head>
  <body>
    <h1>
      服务器文件列表/如下
    </h1>
    <%
      String file=application.getRealPath("/");
      File f=new File(file);
      String[] fileNames=f.list();
      File[] fileObjects=f.listFiles();
    %>
    <table border=0 cellspacing=0 width=100%>
      <tr>
        <td width=30%>文件名</td>
        <td width=70%>大小</td>
      </tr>
    <%
      for (int i=0; i<fileObjects.length; i++) {
        if (!fileObjects[i].isDirectory()) {
    %>
      <tr>
        <td width=30%><a href="<%=fileNames[i]%>"><%=fileNames[i]%></a></td>
        <td width=70%><%=Long.toString(fileObjects[i].length())%>bytes</td>
      <%
        }
      }
      %>
      </tr>
    </table>
  </body>
</html>
```

(2) 保存并在 Tomcat 服务器中测试,检查运行效果是否如图 10-10 所示。

练习 2:读取当前目录下文件和目录列表,并实现目录列表的层次访问。

(1) 打开 JSP 开发环境,修改 simplefilelist.jsp 并保存为 filelist.jsp,在其中加入对目录访问的功能,并实现通过 dir 参数来访问子目录列表。修改方法见程序清单 10-9。

程序清单 10-9:

```
<!--filelist.jsp -->
<%@page contentType="text/html; charset=gb2312" language="java"%>
<%@page import="java.io.*"%>
```



图 10-10 服务器文件列表访问

```
<html>
  <head>
    <title>文件列表</title>
  </head>
  <body>
    <%
      String dir=request.getParameter("dir");
      if(dir==null)
        dir="";
    %>
    <h1>服务器文件列表 /<%=dir%>        如下
    </h1>
    <%
     String file=application.getRealPath("/");
     if(!dir.equals("")){
       file=file+File.separator+dir;
     }
     File f=new File(file);
     String[] fileNames=f.list();
     File[] fileObjects=f.listFiles();
    %>
    <table border="1" cellspacing="1" width="100%">
      <tr>
        <td width="30%">
          文件名
        </td>
        <td width="70%">
          大小
        </td>
      </tr>
      <%
        for (int i=0; i<fileObjects.length; i++) {
        if (!fileObjects[i].isDirectory()) {
```

```
        %>
        <tr>
          <td width="30%">
            <a href="<%=fileNames[i]%>"><%=fileNames[i]%></a>
          </td>
          <td width="70%"><%=Long.toString(fileObjects[i].length())%>bytes
          </td>
          <%
            }else{
              String path="filelist.jsp?dir="+fileNames[i];
          %>
          <td width="30%">
            <a href="<%=path%>">[<%=fileNames[i]%>]</a>
          </td>
          <td width="70">
             
          </td>
          <%
            }
          %>
        </tr>
        <%
          }
        %>
      </table>
    </body>
  </html>
```

(2) 保存并测试 JSP 文件,得到如图 10-11 所示效果,并测试该 JSP 文件是否能按照实验要求工作。



图 10-11 服务器文件及目录的列表

10.4　实验 10.3　JSP 中文件的上传与下载

实验目的：

（1）掌握 JSP 中读写文件的方法。

（2）了解 JSP 中文件上传下载的原理。

（3）了解 jspSmartUpload.jar 上传组件的基本结构和使用方法。

（4）练习 JSP 中文件的上传和在服务器端的保存。

实验内容：

本次实验用于读者掌握 JSP 中文件上传技术，实验模拟了一个简单的学生实验报告上传程序，学生用学号登录，允许查看自己已经上传的实验文件和实现实验文件的上传操作。为了便于管理，在服务器新建 lab 目录，用于保存学生上传的文件，并为每个学生按照登录名建立目录，在其中建立实验 1-4 的子目录 lab1-4。学生的实验文件上传后保存在自己的目录中对应实验子文件夹中。

实验步骤：

（1）打开 Web 开发环境，新建一个文件 login.htm，用于学生的登录，学生按照学号或者姓名登录后定位到对应的个人文件夹中，将程序清单 10-10 输入。

程序清单 10-10：

```
<!--login.htm -->
<html>
<head>
<title>登录页面</title>
</head>

<body>
<h1>实验作业上传<br></h1>
<form name="form1" method="post" action="uploadform.jsp">
  <table border="0" cellspacing="0" cellpadding="0">
    <tr>
      <td height="18" align="right">
        <div align="right">学号：</div></td>
      <td><input name="name" type="text" size="20"></td>
      <td colspan="2">  
        <input type="submit" name="Submit" value="登录">
      </td>
    </tr>
  </table>
</form>
</body>
</html>
```

(2) 保存并测试 html 文件,检查运行效果如图 10-12 所示。



图 10-12　学生实验作业上传登录

(3) 新建 uploadform.jsp 文件,要求实现用户显示登录学生的实验文件上传情况和实现实验文件上传功能。仔细阅读程序清单 10-11,请将 代码 1 ~ 代码 4 补充完整。

程序清单 10-11:

```
<!--uploadform.jsp -->
<%@page contentType="text/html; charset=gb2312" language="java"%>
<%@page import="java.io.*"%>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=gb2312">
    <title>学生实验文件列表</title>
  </head>
  <%
    //读取学生的学号,并存放 session 对象中
    if(request.getParameter("name")!=null){
      session.setAttribute("name", 代码 1 );
    }
    //打开学生的实验对应文件目录
    String file=application.getRealPath("/lab");
    file=file+File.separator+session.getAttribute("name");
    //得到该实验目录下的所有实验文件
    File f= 代码 2 ;
    if(f!=null&&!f.exists())
      f. 代码 3 ;
  %>
  <body>
    <h1><%=session.getAttribute("name")%>同学,您好!
    </h1>
    <hr>
    <table border="1" cellspacing="1" width="600">
      <tr>
```

```
    <td width="200">
      实验名称
    </td>
    <td width="400">
      已经提交实验文件列表
    </td>
  </tr>
  <%
//遍历学生的 4 个实验文件夹
for(int i=1;i<=4;i++){
  String labN="lab"+i;
  File subf=new File(file+File.separator+labN);
  if(subf!=null&&!subf.exists())
    subf.mkdir();
  String[] fileNames=subf.list();
  File[] fileObjects=subf.listFiles();
  out.print("<tr><td>实验"+i+"</td>");
  out.print("<td> ");
  for(int j=0; j<fileObjects.length; j++) {
    if(!fileObjects[j].isDirectory()) {
      out.print("<a href=\"../lab/"+session.getAttribute("name")+"/lab"+i+"/"+
          fileNames[j]+"\">"+fileNames[j]+"</a>"+"  ");
    }
  }
  out.print("</td>");
  out.print("</tr>");
}
%>
</table>
<hr>
<b>实验文件上传</b>
<form method="post" action="upload.jsp"  代码 4 >
  <table border="0" cellspacing="0" cellpadding="0">
    <tr>
      <td>
        选择实验
      </td>
      <td>
        <select name="labName">
          <option value="lab1" selected>
            实验 1
          </option>
          <option value="lab2">
            实验 2
          </option>
```

```
                <option value="lab3">
                  实验 3
                </option>
                <option value="lab4">
                  实验 4
                </option>
              </select>
            </td>
          </tr>
          <tr>
            <td>
              实验报告(*.doc)
            </td>
            <td>
              <input name="file1" type="file" size="40">
            </td>
          </tr>
          <tr>
            <td>
              实验代码(*.rar)
            </td>
            <td>
              <input name="file2" type="file" size="40">
            </td>
          </tr>
          <tr>
            <td colspan="2">
              <input type="submit" name="Submit" value="上传">
            </td>
          </tr>
        </table>
      </form>
    </body>
  </html>
```

(4) 在 uploadform.jsp 学生选择了需要上传文件后,将由 upload.jsp 来实现上传和保存,新建 upload.jsp 文件,将程序清单 10-12 输入。

程序清单 10-12:

```
<!--upload.jsp -->
<%@page language="java" import="com.jspsmart.upload.*"%>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=gb2312">
  </head>
  <jsp:useBean id="mySmartUpload" scope="page"
```

```
    class="com.jspsmart.upload.SmartUpload"/>
  <body>
  <%
      try {
          int count=0;
          mySmartUpload.initialize(pageContext);
          mySmartUpload.setMaxFileSize(10000000);
          mySmartUpload.setTotalMaxFileSize(10000000);
          mySmartUpload.setAllowedFilesList("doc,rar");
          mySmartUpload.setDeniedFilesList("exe,bat");
          mySmartUpload.upload();
          String stu=(String)session.getAttribute("name");
        String labN=mySmartUpload.getRequest().getParameter("labName");
          for (int i=0; i<mySmartUpload.getFiles().getCount(); i++) {
              File myFile=mySmartUpload.getFiles().getFile(i);
              if (!myFile.isMissing()) {
                myFile.saveAs("/lab/"+stu+"/"+labN+"/"+myFile.getFileName(),
                SmartUpload.SAVE_VIRTUAL);
              }
          }
          response.sendRedirect("uploadform.jsp");
      } catch (Exception e) {
          out.println(e);
      }
  %>
  </body>
</html>
```

(5) 保存并在 Tomcat 服务器中测试,检查是否能得到如图 10-13 的效果。

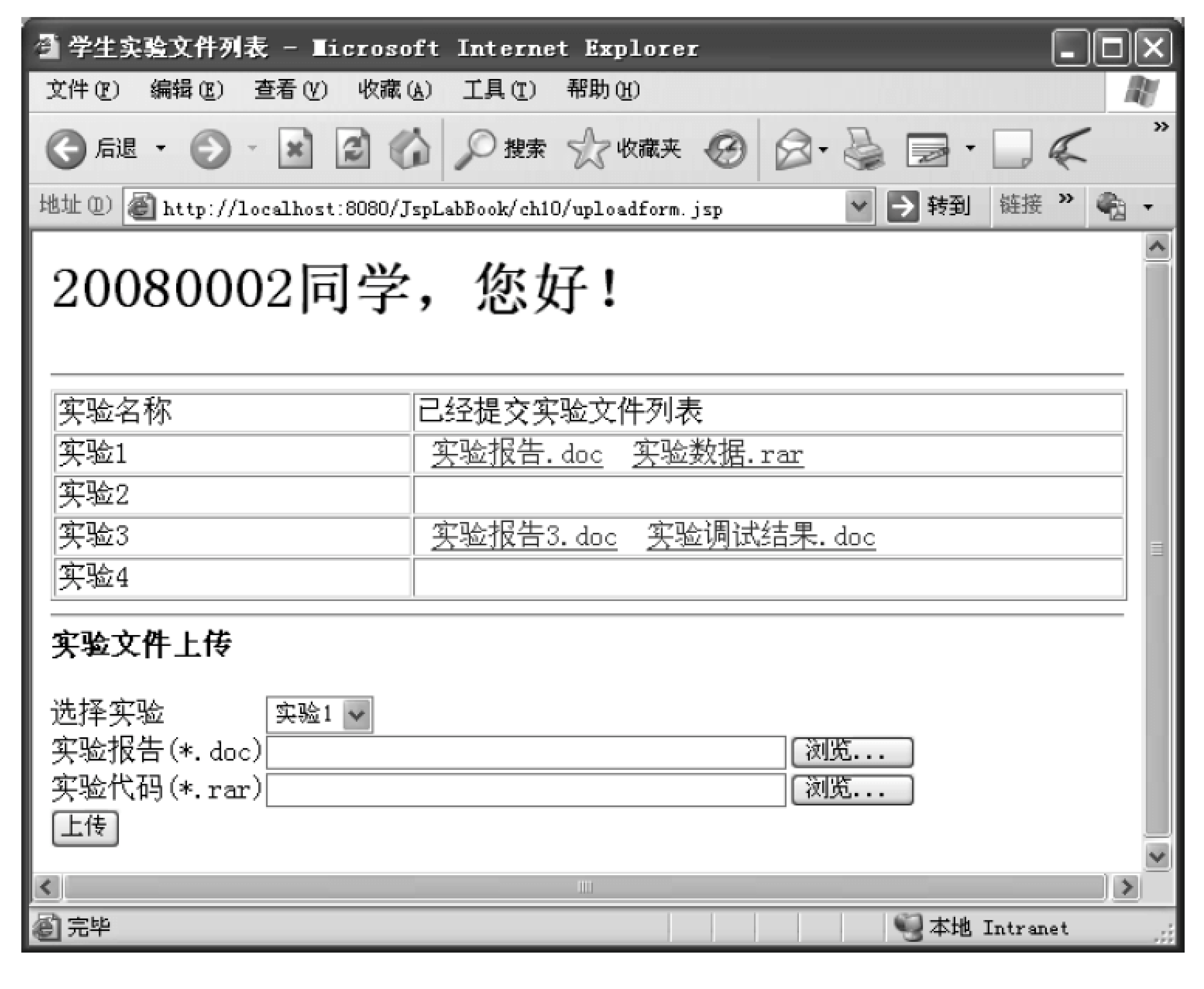


图 10-13　学生实验文件列表和上传

第 11 章 JSP 访问 Web 数据库

JSP 开发离不开数据库编程，几乎所有的 JSP 开发都和数据库有关，JSP 中提供了通过 JDBC 接口访问数据库的方法，大大简化了 JSP 数据库开发的难度。本章将通过几个实验帮助读者掌握 JSP 中访问数据库的方法。

11.1 预备知识

11.1.1 JDBC 基本概念

JDBC(Java DataBase Connectivity)是一种用 Java 实现的数据库接口技术，是开放数据库 ODBC 的 Java 实现。用 JDBC 开发数据库应用的原理如图 11-1 所示。

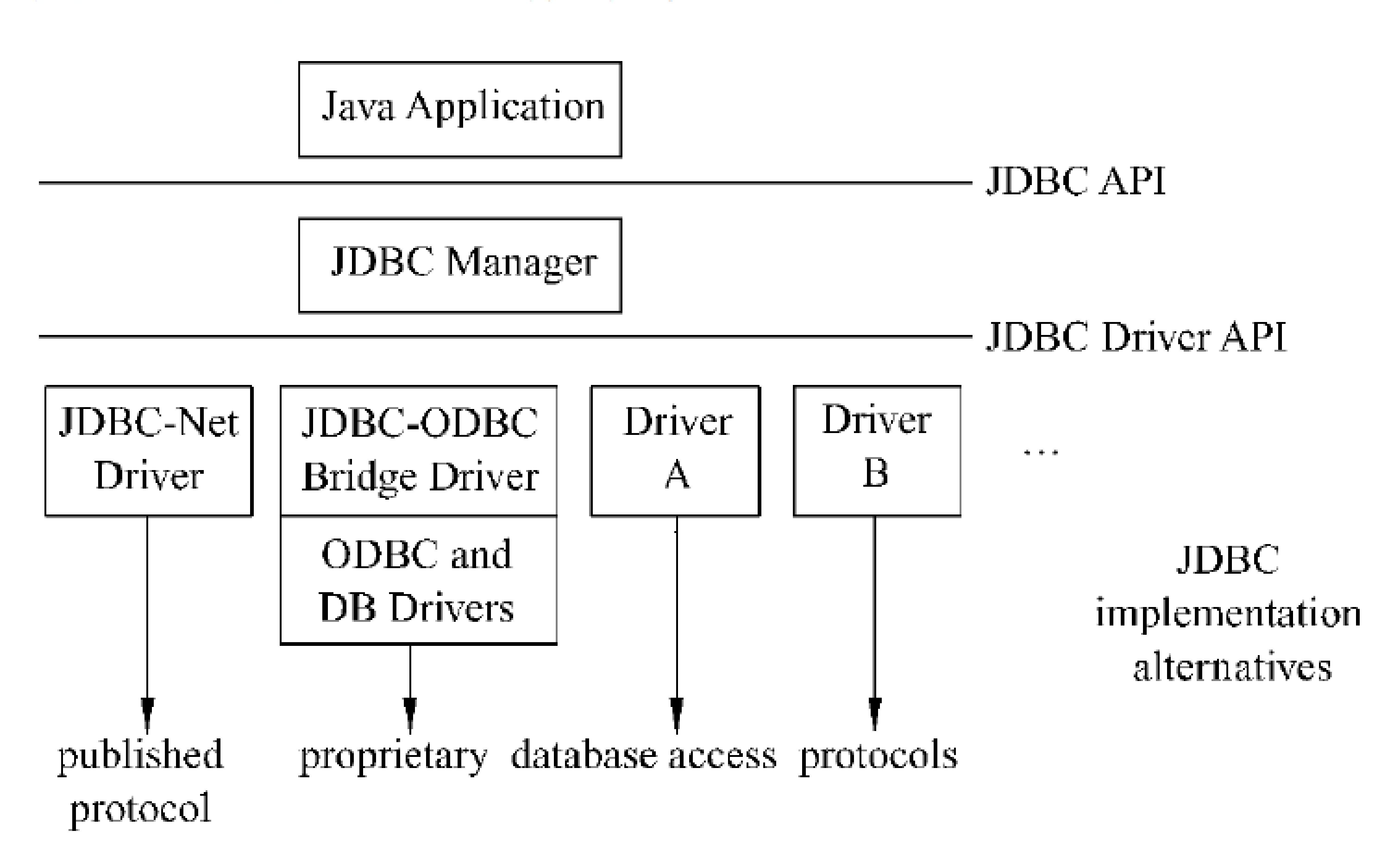


图 11-1 JDBC 工作原理

由图 11-1 可知，JDBC 由两层组成。上面一层是 JDBC API，负责与 Java 应用程序通信，向 Java 应用程序提供数据(Java 应用程序通过 JDBC 中提供的相关类来管理 JDBC 的驱动程序)。下面一层是 JDBC Driver API，主要负责和具体数据环境的连接。

11.1.2 数据库的连接方式

总体来说，有 4 种数据库连接的方式。

(1) JDBC-ODBC 桥。通过 JDBC-ODBC 桥，可以将 JDBC 对数据库的操作映射为 ODBC 对于数据库的操作。

(2) 部分 Java，部分数据库专用 API。这种方式使用 Java 实现和数据库专用 API 混合方式连接。JDBC 驱动将标准的 JDBC 调用转化为对数据库 API 的本地调用。

(3) 中间件访问。这种访问方式将 JDBC 的操作指令转换为驱动程序厂商自己定义的网络协议，该网络协议是与 DBMS 无关的。通过协议访问某种特定的中间件服务器。这个特定的中间件服务器往往位于 Web 服务器或者数据库服务器上。然后由这个中间件服务

器将网络协议再转换为某种特定的 DBMS 协议，来调用数据库。数据库处理的结果也将按照相反的过程返回给 Java 程序端。

（4）纯 Java 驱动访问。这种访问方式使用厂商专用的驱动程序，把 JDBC 对数据库的操作直接转换为针对某种数据库进行操作的本地协议。

11.1.3 JDBC 常用接口

JDBC 定义了很多的接口种类，在 JDBC API 中对数据库的应用主要是对 DriverManager、Connection、Statement 和 ResultSet 这几个类和接口的使用，它们的调用结构如图 11-2 所示，在本节中，将对这个几个类和接口做简要的说明。

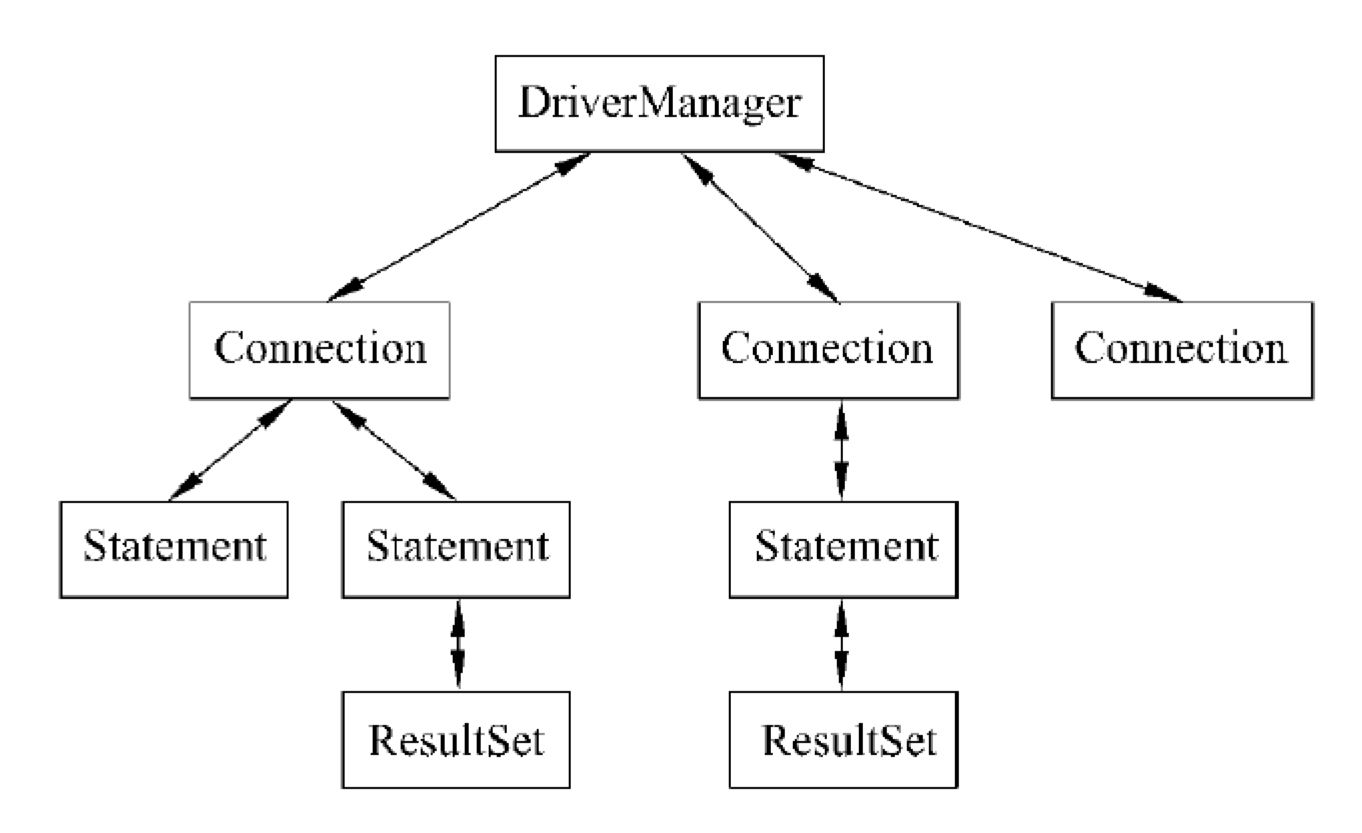


图 11-2 JDBC 主要接口的调用结构

1. DriverManager 类

DriverManager 类是 JDBC 的管理层，作用于用户和驱动程序之间。它跟踪可用的驱动程序，并在数据库和相应驱动程序之间建立连接。

如下代码所示：

```
try{
  //装载 MySQL 5.0 驱动
  Class.forName("com.mysql.jdbc.Driver").newInstance();
  //装载 SQL Server 5.0 驱动
  Class.forName("com.microsoft.jdbc.sqlserver.SQLServerDriver ").newInstance();
  //装载 SQL Server 2005 驱动
  Class.forName("com microsoft.sqlserver.jdbc.SQLServer"
  //装载 Oracle 驱动
  Class.forName("oracle.jdbc.driver.OracleDriver").newInstance();
catch(ClassNotFoundException e){
  //若装载失败，则抛出 ClassNotFoundException
}
```

2. Connection 接口

Connection 是与数据库的连接对象，也就是在已经加载的 Driver 和数据库之间建立连接。

如下代码所示：

```
try{
  //根据 URL 建立连接
```

```
    Connection conn=DriverManager.getConnection(url,user,password);
    }catch(SQLException e){
    //若建立连接失败,则抛出 SQLException
  }
```

3. Statement 接口

Statement 用于将 SQL 语句发送到数据库中,并获取指定 SQL 语句的结果。

Statement 对象由 Connection 对象的 createStatement()方法创建,如下列代码所示:

```
Connection con=DriverManager.getConnection(url, "root","");
Statement stmt=con.createStatement();
```

Statement 对象用于执行不带参数的简单 SQL 语句,它的典型使用是用 executeQuery 执行查询,用 executeUpdate 执行更新,如下代码:

```
Statement stmt=con.createStatement();
//以下代码执行从表 employee 中查询所有内容
ResultSet rs=stmt.executeQuery("select * from employee");
//以下代码执行删除表 employee 中一条记录
int changedLine=stmt.executeUpdate("delete from employee where id='0005'");
```

4. ResultSet 类

ResultSet 结果集对象负责存储数据库查询的结果,并提供一系列方法对数据库进行增加、删除和修改操作。

11.2　实验 11.1　JSP 中数据库的连接

实验目的:

(1) 掌握 JDBC 的基本结构和主要接口使用方法。
(2) 学会在 JSP 中连接数据库的方法。
(3) 能够处理 JSP 连接数据库时可能出现的问题和异常。
(4) 学会使用 SQLException 类。

实验内容:

本次实验训练读者掌握 JSP 连接数据库的技术,实验模拟了数据库的连接,连接的相关内容由用户输入。通过这次实验可以了解到数据库连接可能出现的问题和原因,帮助以后开发正确的数据库程序。

实验步骤:

(1) 本实验中需要用到 MySQL 5.0、SQL Server 2005、Oracle log 数据库,安装并调试以上数据库系统,建立相应的测试数据库。下载对应数据库的 JDBC 驱动并在测试环境中安装和设置好环境变量。

(2) 打开 JSP 开发环境,新建一个 JSP 文件 testconn.jsp,用于实现对各种数据库的连

接，将程序清单 11-1 中的代码输入。

程序清单 11-1：

```
<!--testconn.jsp-->
<%@page contentType="text/html;charset=gb2312"%>
<%@page import="java.sql.*"%>
<html>
  <head>
    <title>数据库连接测试</title>
  </head>
  <body>
    <%
      String btConn=request.getParameter("conn");
      String driver="";
      String url="";
      if(btConn!=null){
        String dbms=request.getParameter("dbms");
        String db=request.getParameter("database");
        String user=request.getParameter("user");
        String pass=request.getParameter("pass");
        if(dbms.equals("mysql")){
          driver="com.mysql.jdbc.Driver";
          url="jdbc:mysql://localhost/"+db;
        }else if(dbms.equals("sqlserver")){
          driver="com.microsoft.sqlserver.jdbc.SQLServerDriver";
          url="jdbc:sqlserver://localhost:1433;DatabaseName="+db;
        }else if(dbms.equals("oracle")){
          driver="oracle.jdbc.driver.OracleDriver";
          url="jdbc:oracle:thin:@localhost:1521:"+db;
        }
        //尝试连接
        try {
          Class.forName(driver).newInstance();
          Connection conn=DriverManager.getConnection(url, user, pass);
          if (conn!=null){
            out.print("<h3>数据库连接成功,恭喜你!</h3>");
            out.print("<p>数据库驱动程序"+driver+"</p>");
            out.print("<p>连接字符串是"+url+"</p>");
          }
          conn.close();
        } catch (SQLException sqle) {
          out.print(sqle.getSQLState()+" ");
          out.print(sqle.getMessage());
          out.print("<p>数据库驱动程序 "+driver+"</p>");
          out.print("<p>连接字符串是 "+url+"</p>");
```

```
        } catch(Exception e){
          out.print(e.toString());
        }
        out.print("<hr>");
      }

    %>
    <!--用户连接选择-->
    <form action=testconn.jsp method="post">
      <table border="0" cellspacing="0">
        <tr>
          <td>数据库系统：</td>
          <td>
            <select name="dbms">
              <option value="mysql">MySQL</option>
              <option value="sqlserver" selected>SQL Server 2005</option>
              <option value="oracle">Oracle</option>
            </select>
          </td>
        </tr>
        <tr>
          <td>连接数据库：</td>
          <td><input type="text" size=30 name="database"></td>
        </tr>
        <tr>
          <td>用户名：</td>
          <td><input type="text" size=30 name="user"></td>
        </tr>
        <tr>
          <td>密码：</td>
          <td><input type="password" size=30 name="pass"></td>
        </tr>
       </table>
      <input type="submit" value="连接" name="conn">
    </form>
  </body>
</html>
```

(3) 保存 testconn.jsp，分别启动上述 3 个数据库服务，启动 Tomcat 服务器，测试 testconn.jsp，运行效果如图 11-3 所示。

(4) 输入一个正确的数据库名、用户名和密码，单击“连接”按钮，程序将显示成功连接到数据库，并反馈给用户连接驱动程序和连接字符串等信息，如图 11-4 所示。

(5) 分别输入错误的数据库名、用户名、密码测试，程序将无法连接到数据库，并反馈错误的基本信息。例如，输入一个在数据库中并不存在的数据库 mytt，观察并记录运行结果。又如输入错误的用户名 user1，观察并记录运行结果，并解释原因。



图 11-3 数据库连接测试

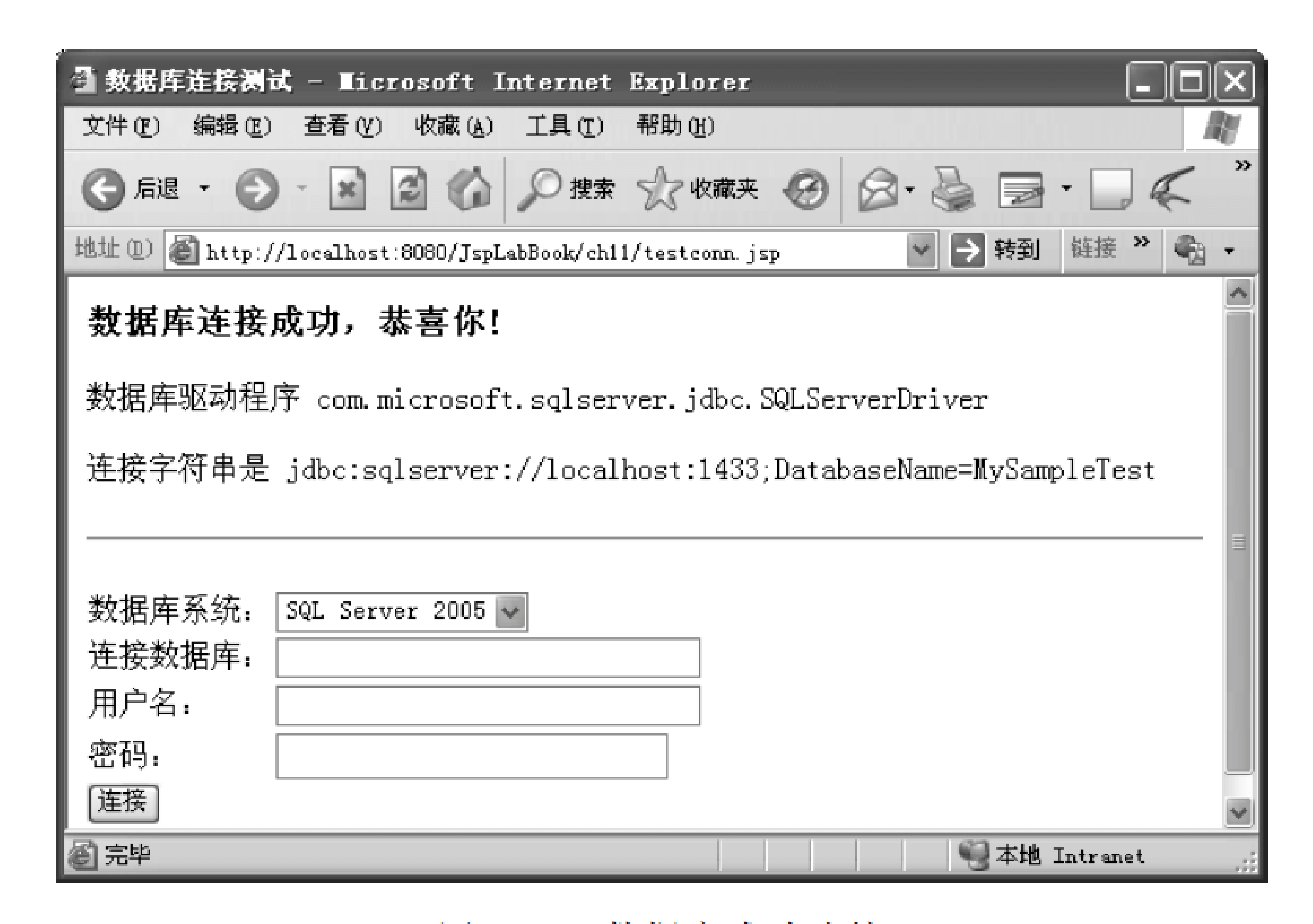


图 11-4 数据库成功连接

11.3 实验 11.2 数据库查询

实验目的：

（1）掌握 JDBC 的基本结构和主要接口使用方法。

（2）学会在 JSP 中使用 JDBC 接口。

（3）掌握 JSP 中使用 SQL 语言查询数据库的方法。

实验内容：

本次实验训练读者掌握 JSP 数据库的查询技术，涉及 JDBC 中的相关接口和 SQL 语言。本次实验编写一个数据读取的 JSP 文件，读取数据库中表格的格式和数据。实验由 3 个练习构成。

（1）查询数据库中表格的格式，将表格的字段列表显示。

（2）查询数据库中的表格，将表格数据列表显示。

(3) 利用后台 JavaBean 实现数据库的查询。

实验步骤:

练习 1: 查询数据库中表格的格式,将表格的字段列表显示。

(1) 本次实验需要后台数据库的支持,安装并调试好数据库系统,实验中使用 SQL Server 2005 作为数据库环境。下载 SQL Server 2005 的 JDBC 驱动 sqljdbc.jar,并复制到 Tomcat 服务器的 lib 目录或者具体 Web 应用程序的 WEB-INF/lib 目录中。

(2) 打开 SQL Server 2005 的客户端管理工具 SQL Server Management Studio,登录后如图 11-5 所示。

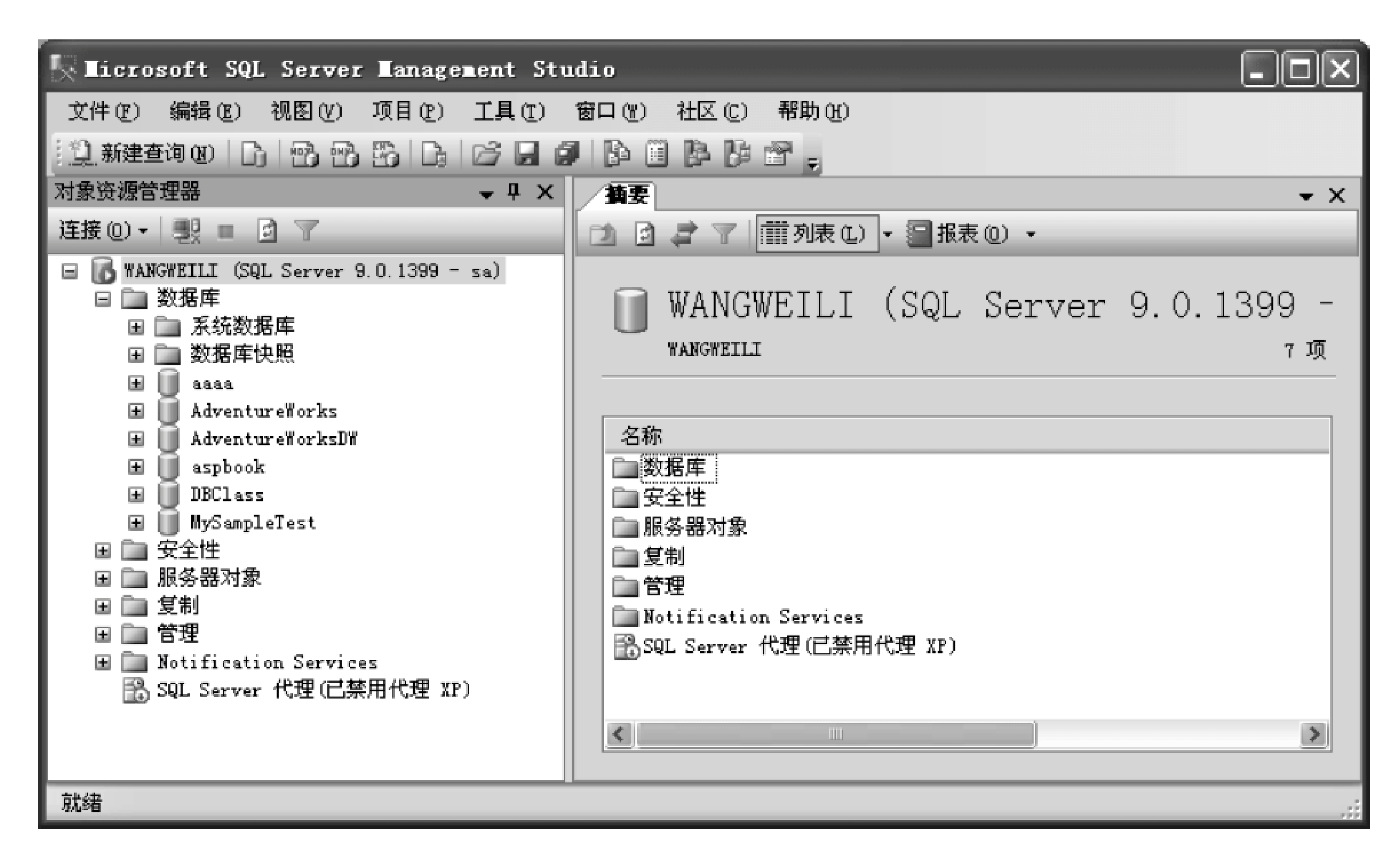
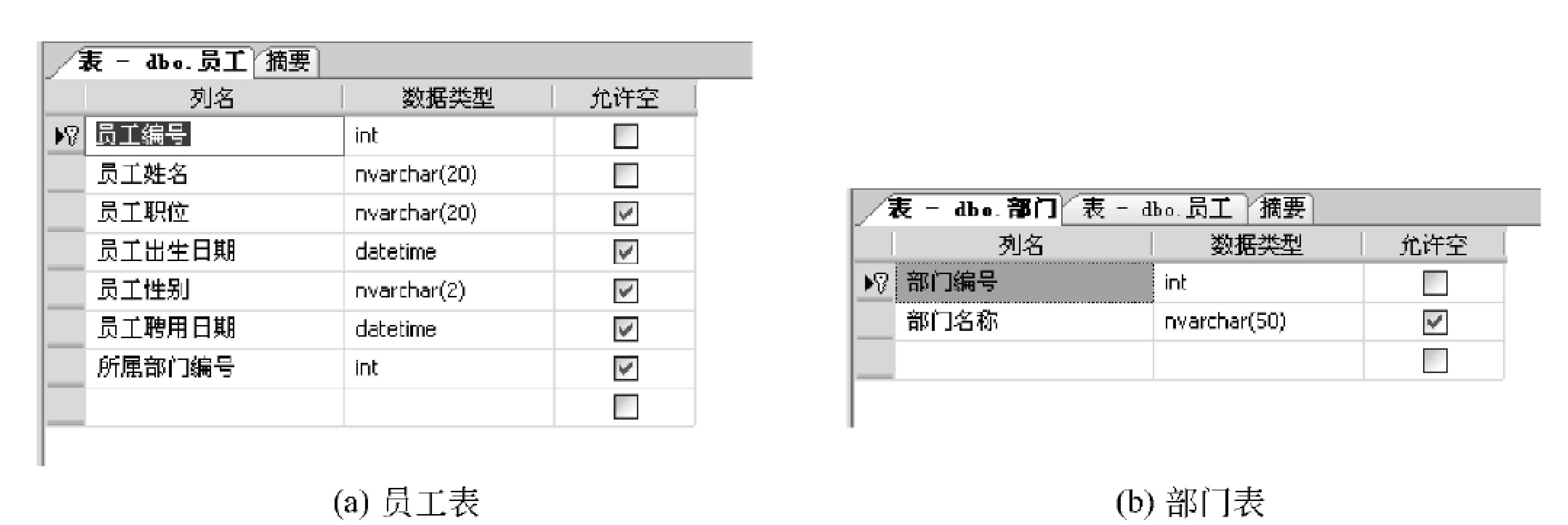


图 11-5 SQL Server Management Studio

(3) 建立数据库 MySampleTest,并在其中建立两个表"员工"和"部门",其中如图 11-6(a)所示员工表的"所属部门编号"数据是外键参考了部门表 11-6(b)中的"部门编号"属性,如图 11-6 所示。并在表中插入测试用数据。



(a) 员工表　　　　(b) 部门表

图 11-6 实验用测试数据库和表格

注意: 读者也可以根据实际情况,采用其他数据库,如 MySQL 以及相应的 JDBC 驱动来实现下列的实验练习。

(4) 打开 JSP 开发环境,新建 JSP 文件 employeemetadate.jsp,部分代码见程序清

单 11-2,该文件用于读取数据库 MySampleTest 中"员工"表的结构。请仔细阅读程序,将程序清单 11-2 中的 代码 1 ～ 代码 4 处填充完整,实现对数据库"员工表"结构的显示。

程序清单 11-2:

```
<!--employeemetadate.jsp -->
<%@page contentType="text/html; charset=gb2312" language="java"%>
<%@page import="java.sql.*"%>
<html>
  <head>
    <title>员工表</title>
  </head>
  <%
   try{
    Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver").newInstance();
    String url="jdbc:sqlserver://localhost:1433;DatabaseName=MySampleTest";
    Connection conn= 代码 1 ;
    //建立连接对象
    Statement stmt= 代码 2 ;
    //建立查询对象
  %>
  <body>
    员工表结构如下
    <br>
    <table border="1" cellspacing="1">
      <tr>
        <td>字段名称</td>
        <td>类型</td>
        <td>列宽</td>
      </tr>
      <%
        String sql="select * from 员工";
        ResultSet rs= 代码 3 ;
        //执行查询操作
        ResultSetMetaData meta=rs.getMetaData();
        for(int i=1;i<=meta.getColumnCount();i++){
          out.print("<tr><td>"+meta.getColumnName(i)+"</td>");
          out.print("<td>"+meta.getColumnTypeName(i)+"</td>");
          out.print("<td>"+meta.getPrecision(i)+"</td></tr>");
       }
    %>
    </table>
  </body>
  <%
    rs.close();
```

```
      stmt.close();
      conn.close();
    }catch(代码 4){
       //捕获驱动程序不存在的异常
       out.print(cnfe);
    }catch(SQLException sqle){
      out.print(sqle);
    }catch(Exception e){
      out.print(e);
    }
%>
</html>
```

(5) 修改后保存 employeemetadate.jsp 文件,启动 Tomcat 服务器,测试该文件是否能正确读取“员工”表格的格式,使之运行效果如图 11-7 所示。



图 11-7 “员工”表格式

练习 2:查询数据库中的表格,将表格数据列表显示。

(1) 建立 JSP 文件 employeeR.jsp,用于实现员工信息的显示。由于员工的所属部门编号需要参考部门表,所以在程序中需要用到连接查询的 SQL 语句。employeeR.jsp 的部分代码见程序清单 11-3,请将程序清单 11-3 中的 代码 1 ~ 代码 3 补充完整。注意:在连接数据库的用户名和密码要与数据库设置保持一致。

程序清单 11-3:

```
<!--employeeR.jsp -->
<%@page contentType="text/html; charset=gb2312" language="java"%>
<%@page import="java.sql.*"%>
<html>
  <head>
    <title>员工表</title>
```

```
</head>
<%
 try{
  Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver").newInstance();
  String url="jdbc:sqlserver://localhost:1433;DatabaseName=MySampleTest";
  Connection conn=DriverManager.getConnection(url,"user1","passw1");
                                   //其中 user1 为用户名,passw1 为密码名,由用户自行设置
  Statement stmt=conn.createStatement();
%>
<body>
  员工信息如下
  <br>
  <table border="1" cellspacing="1">
    <tr>
      <td>员工编号</td>
      <td>员工姓名</td>
      <td>员工职位</td>
      <td>员工出生日期</td>
      <td>员工性别</td>
      <td>部门名称</td>
    </tr>
    <%
      String sql="select 员工编号,员工姓名,员工职位,员工出生日期,员工性别,部门名称 ";
      sql=sql+"from 员工 left outer join 部门 on (员工.所属部门编号=部门.部门编号)";
      ResultSet rs=stmt.代码 1;
      //执行 sql 查询
      while(rs!=null && 代码 2){
       //依次输出查询结果
        out.print("<tr><td>"+rs.getInt(1)+"</td>");
        out.print("<td>"+rs.getString(2)+"</td>");
        out.print("<td>"+rs.getString(3)+"</td>");
        out.print("<td>"+rs.代码 3+"</td>");
        out.print("<td>"+rs.getString(5)+"</td>");
        out.print("<td>"+rs.getString(6)+"</td></tr>");
     }
   %>
  </table>
</body>
<%
  rs.close();
  stmt.close();
  conn.close();
}catch(ClassNotFoundException cnfe){
    out.print(cnfe);
}catch(SQLException sqle){
  out.print(sqle);
```

```
    }catch(Exception e){
      out.print(e);
    }
%>
</html>
```

（2）保存并测试 employeeR.jsp，检查程序是否返回了员工的数据，如图 11-8 所示。



图 11-8 “员工”表数据

练习 3：利用后台 JavaBean 实现数据库的查询。

（1）利用后台 JavaBean 实现对数据表的查询。打开 Java 开发环境，新建 Java 文件 Employee.java，在该类中实现数据库的连接和数据的查询。将程序清单 11-4 的代码输入。

程序清单 11-4：

```
//Employee.java
import java.text.*;
import java.util.Date;
import java.sql.*;
import java.io.*;

public class Employee {

    public Connection getConnection() {
        Connection conn=null;
        try {
            Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver")
                    .newInstance();
            String url="jdbc:sqlserver://localhost:1433;DatabaseName=MySampleTest";
            conn=DriverManager.getConnection(url, "user1", "passw1");
        } catch (Exception e) {
        }
        return conn;
    }
```

```
    public ResultSet readEmployees() {
        try {
            Statement stmt=getConnection().createStatement();
            String sql="select 员工编号,员工姓名,员工职位,员工出生日期,员工性别,部门名称 ";
            sql=sql+"from 员工 left outer join 部门 on(员工.所属部门编号=部门.部门编号)";
            ResultSet rs=stmt.executeQuery(sql);
            return rs;
        } catch (Exception e) {
        }
        return null;
    }
}
```

(2) 新建 JSP 文件 employeewithbean.jsp,利用程序清单 11-4 中的 JavaBean 完成员工表数据的读取,将程序清单 11-5 中的代码输入。

程序清单 11-5:

```
<!--employeewithbean.jsp -->
<%@page contentType="text/html; charset=gb2312" language="java"%>
<%@page import="java.sql.*"%>
<jsp:useBean id="employee" class="chapter11.Employee" scope="page"/>
<html>
  <head>
    <title>员工表</title>
  </head>
  <body>
    员工信息如下
    <br>
    <table border="1" cellspacing="1">
      <tr>
        <td>员工编号</td>
        <td>员工姓名</td>
        <td>员工职位</td>
        <td>员工出生日期</td>
        <td>员工性别</td>
        <td>部门名称</td>
      </tr>
      <%
        ResultSet rs=employee.readEmployees();
        while(rs!=null && rs.next()){
          out.print("<tr><td>"+rs.getInt(1)+"</td>");
          out.print("<td>"+rs.getString(2)+"</td>");
          out.print("<td>"+rs.getString(3)+"</td>");
          out.print("<td>"+rs.getDate(4)+"</td>");
          out.print("<td>"+rs.getString(5)+"</td>");
          out.print("<td>"+rs.getString(6)+"</td></tr>");
```

```
        }
      %>
      </table>
    </body>
    <%
      rs.close();
    %>
</html>
```

(3) 保存并调试 employeewithbean.jsp 程序，看是否和直接用 JSP 脚本实现的效果一致。

11.4 实验 11.3 数据库更新

实验目的：

(1) 掌握 JDBC 的基本结构和主要接口使用方法。
(2) 学会在 JSP 中使用 JDBC 接口。
(3) 掌握 JSP 中使用 SQL 语言更新数据库的方法。

实验内容：

本次实验训练读者掌握 JSP 数据库的更新技术。数据库的更新主要有插入、修改和删除操作，涉及到 SQL 语言中的 insert、update 和 delete 操作。本次实验编写一个数据更新程序，还是用到实验 11.2 中的数据库，实现对员工数据的更新。

实验步骤：

本次实验的具体要求如下：修改实验 11.2 中程序清单 11-3 的 employeeR.jsp，并保存为新的 JSP 文件 employeeRW.jsp，加入新建一个员工、修改和删除员工的链接。新建员工和修改员工都是利用一个表单来实现的，链接到 employeeform.jsp，依靠 action 参数来区分这两个操作。如果 action 参数的值为 new，则表示为“新建一个员工”；如果 action 参数的值为 edit，表示“修改员工信息”。删除员工从表单，直接链接到 employeedelete.jsp 即可。

(1) 请仔细阅读程序清单 11-6，将程序 employeeRW.jsp 中的 代码 1 ～ 代码 3 补充完整，实现实验练习的要求，运行结果如图 11-9。

程序清单 11-6：

```
<!--employeeRW.jsp -->
<%@page contentType="text/html; charset=gb2312" language="java"%>
<%@page import="java.sql.*"%>
<html>
  <head>
    <title>员工表</title>
  </head>
```

```
<%
 try{
  Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver").newInstance();
  String url="jdbc:sqlserver://localhost:1433;DatabaseName=MySampleTest";
  Connection conn=DriverManager.getConnection(url,"user1","passw1");
  Statement stmt=conn.createStatement();
%>

<body>
  员工信息如下
  代码 1
  <!--新建员工链接-->
  <br>
  <table border="1" cellspacing="1">
    <tr>
      <td>员工编号</td>
      <td>员工姓名</td>
      <td>员工职位</td>
      <td>员工出生日期</td>
      <td>员工性别</td>
      <td>部门名称</td>
    </tr>
    <%
      String sql="select 员工编号,员工姓名,员工职位,员工出生日期,员工性别,部门名称 ";
      sql=sql+"from 员工 left outer join 部门 on (员工.所属部门编号=部门.部门编号)";
      ResultSet rs=stmt.executeQuery(sql);
      while(rs!=null && rs.next()){
        out.print("<tr><td>"+rs.getInt(1)+"</td>");
        out.print("<td>"+rs.getString(2)+"</td>");
        out.print("<td>"+rs.getString(3)+"</td>");
        out.print("<td>"+rs.getDate(4)+"</td>");
        out.print("<td>"+rs.getString(5)+"</td>");
        out.print("<td>"+rs.getString(6)+"</td>");
        out.print("<td>");
        代码 2;
        //修改链接
        out.println("</td>");
        out.print("<td>");
        代码 3;
        //删除链接
        out.println("</td>");
     }
   %>
  </table>
```

```
  </body>
  <%
    rs.close();
    stmt.close();
    conn.close();
  }catch(ClassNotFoundException cnfe){
      out.print(cnfe);
  }catch(SQLException sqle){
    out.print(sqle);
  }catch(Exception e){
    out.print(e);
  }
  %>
</html>
```



图 11-9　可更新“员工”表数据

(2) 新建 JSP 文件 employeeform. jsp。这个文件用于接收新建员工数据或修改的员工数据,通过加入 HTML 表单来实现,将程序清单 11-7 的代码输入:

程序清单 11-7:

```
<!--employeeform.jsp -->
<%@page contentType="text/html; charset=gb2312" language="java"%>
<%@page import="java.sql.*"%>
<html>
  <head>
    <title>员工管理</title>
  </head>
  <%
    Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver").newInstance();
    String url="jdbc:sqlserver://localhost:1433;DatabaseName=MySampleTest";
    Connection conn=DriverManager.getConnection(url,"user1","user1");
    Statement stmt=conn.createStatement();
    ResultSet rs=stmt.executeQuery("select * from 员工");
    String action=request.getParameter("action");
```

```
    String actiontype="";
    String actionaim="";
    String editemployeeid="";
    if(action.equals("new")){
      actiontype="新建一个员工";
      actionaim="employeeadd.jsp";
    }else{
      actiontype="修改员工信息";
      actionaim="employeeedit.jsp";
      editemployeeid=request.getParameter("id");
      rs=stmt.executeQuery("select * from 员工 where 员工编号="+editemployeeid);
      if(rs!=null)rs.next();
    }
  %>
  <body>
    <%=actiontype%>
    <hr>
    <form action="<%=actionaim%>" method="post">
      <table border="0" cellspacing="0">
        <tr>
          <td>员工编号:</td>
          <td>
          <%
            if(action.equals("new"))
              out.print("<input type=\"text\" size=\"10\" name=\"empNo\">");
            else
              out.print("<input type=\"text\" size=\"10\" name=\"empNo\" readonly=true
                value="+rs.getInt("员工编号")+">");
          %>
          </td>
        </tr>
        <tr>
          <td>员工姓名:</td>
          <td>
          <%
           if(action.equals("new"))
             out.print("<input type=\"text\" size=\"20\" name=\"empName\">");
           else
             out.print("<input type=\"text\" size=\"20\" name=\"empName\" value=\""+
               rs.getString("员工姓名")+"\">");
          %>
          </td>
        </tr>
        <tr>
          <td>员工职位:</td>
```

```
  <td>
  <%
   if(action.equals("new"))
     out.print("<input type=\"text\" size=\"20\" name=\"empJob\">");
  else
    out.print("<input type=\"text\" size=\"20\" name=\"empJob\" value=\""+
      rs.getString("员工职位")+"\">");
 %>
  </td>
</tr>
<tr>
  <td>员工出生日期:</td>
  <td>
  <%
   if(action.equals("new"))
     out.print("<input type=\"text\" size=\"20\" name=\"empBir\">");
   else
     out.print("<input type=\"text\" size=\"20\" name=\"empBir\" value=\""+
      rs.getDate("员工出生日期")+"\">");
  %>
  </td>
</tr>
<tr>
  <td>员工性别:</td>
  <td>
    <select name="empSex">
    <%
      if(action.equals("edit")&&rs.getString("员工性别").equals("女")){
        out.print("<option value='男'>男</option>");
        out.print("<option value='女' selected>女</option>");
      }else{
        out.print("<option value='男' selected>男</option>");
        out.print("<option value='女'>女</option>");
      }
    %>
    </select>
  </td>
</tr>
<tr>
  <td>所属部门:</td>
  <td>
    <select name="empDepart">
    <%
      int departno=0;
      if(action.equals("edit"))
```

```
                departno=rs.getInt("所属部门编号");
              ResultSet temprs=stmt.executeQuery("select * from 部门");
              while(temprs!=null && temprs.next()){
                out.print("<option value=\""+temprs.getInt("部门编号")+"\"");
                if(action.equals("edit")&& departno==temprs.getInt("部门编号"))
                  out.print("selected");
                out.print(">"+temprs.getString("部门名称")+"</option>");
              }
              temprs.close();
            %>
            </select>
          </td>
        </tr>
      </table>
      <input type="submit" value="提交">
    </form>
  </body>
  <%
    rs.close();
    stmt.close();
    conn.close();
  %>
</html>
```

（3）在本次实验中利用后台 JavaBean 来实现数据的插入、修改和删除，所以修改程序清单 11-4 中的 Employee.java 代码，加入实现代码。为了方便，将程序清单 11-7 中 employeeform.jsp 的表单在 Employee.java 中加入每个输入对应的参数和 get-set 方法。将 Employee.java 修改为程序清单 11-8 中代码所示。

程序清单 11-8：

```
//Employee.java
import java.text.*;
import java.util.Date;
import java.sql.*;
import java.io.*;

public class Employee {
    private int empNo;
    private String empName;
    private String empJob;
    private String empBir;
    private String empSex;
    private int empDepart;

    public int getEmpNo() {
```

```
//获取编号
    return empNo;
}

public void setEmpNo(int empNo) {
//设置编号
this.empNo=empNo;
}

public String getEmpName() {
 //获取姓名
    return empName;
}

public void setEmpName(String empName) {
 //设置姓名
    this.empName=empName;
}

public String getEmpJob() {
//获取工作性质
    return empJob;
}

public void setEmpJob(String empJob) {
 //设置工作性质
    this.empJob=empJob;
}

public String getEmpBir() {
 //获取出生日期
    return empBir;
}

public void setEmpBir(String empBir) {
 //设置出生日期
    this.empBir=empBir;
}

public String getEmpSex() {
 //获取性别
    return empSex;
}

public void setEmpSex(String empSex) {
```

```
 //设置性别
     this.empSex=empSex;
}

public int getEmpDepart() {
 //获取部门
     return empDepart;
}

public void setEmpDepart(int empDepart) {
 //设置部门
     this.empDepart=empDepart;
}

public String trans(String origin) {
 //中文字符处理
     String result=null;
     try {
         byte[] temp=origin.getBytes("iso-8859-1");
         result=new String(temp, "gb2312");
     } catch (UnsupportedEncodingException usee) {
     }
     return result;
}

public Connection getConnection() {
      //省略,见程序清单 11-4
}

public ResultSet readEmployees() {
     //省略,见程序清单 11-4
}

public void insert() {
 //插入记录处理
     try {
         String sql="insert into 员工 "
         sql=sql+"(员工编号,员工姓名,员工职位,员工出生日期,员工性别,所属部门编号)";
         sql=sql+"values(?,?,?,?,?,?)";
         PreparedStatement pstmt=getConnection().prepareStatement(sql);
         pstmt.setInt(1, Integer.valueOf(empNo).intValue());
         pstmt.setString(2, trans(empName));
         pstmt.setString(3, trans(empJob));
         SimpleDateFormat dd=new SimpleDateFormat("yyyy-MM-dd");
         Date date=dd.parse(empBir);
```

```
                pstmt.setDate(4, new java.sql.Date(date.getTime()));
                pstmt.setString(5, trans(empSex));
                pstmt.setInt(6, empDepart);
                pstmt.executeUpdate();
            } catch (Exception e) {
            }
        }

        public void update() {
       //修改记录
            try {
            String sql="update 员工 ";
            sql=sql+"set 员工姓名=?,员工职位=?,员工出生日期=?,员工性别=?,所属部门编号=?";
                sql=sql+"where 员工编号="+empNo;
                PreparedStatement pstmt=getConnection().prepareStatement(sql);
                pstmt.setString(1, trans(empName));
                pstmt.setString(2, trans(empJob));
                SimpleDateFormat dd=new SimpleDateFormat("yyyy-MM-dd");
                Date date=dd.parse(empBir);
                pstmt.setDate(3, new java.sql.Date(date.getTime()));
                pstmt.setString(4, trans(empSex));
                pstmt.setInt(5, empDepart);
                pstmt.executeUpdate();
            } catch (Exception e) {
            }
        }

        public void delete(String empno) {
        //删除记录
            try {
                String sql="delete from 员工 where 员工编号="+empno;
                Statement stmt=getConnection().createStatement();
                stmt.executeUpdate(sql);
            } catch (Exception e) {
            }
        }
    }
```

(4) 如程序清单 11-6 中代码 employeeRW.jsp 所示，员工新建功能发送到 employeeadd.jsp 处理，员工修改功能发送到 employeeedit.jsp 处理，员工删除发送到 employeedelete.jsp 处理。请仔细阅读下列的程序，将程序清单 11-9～11-11 中的 代码段 1 ～ 代码段 3 补充完整。

程序清单 11-9：

```
<%--employeeadd.jsp--%>
<jsp:useBean id="employee" class="chapter11.Employee" scope="page">
```

```
<jsp:setProperty name="employee" property="*"/>
</jsp:useBean>
<%
  代码段 1;
  //插入记录
  response.sendRedirect("employeeRW.jsp");
%>
```

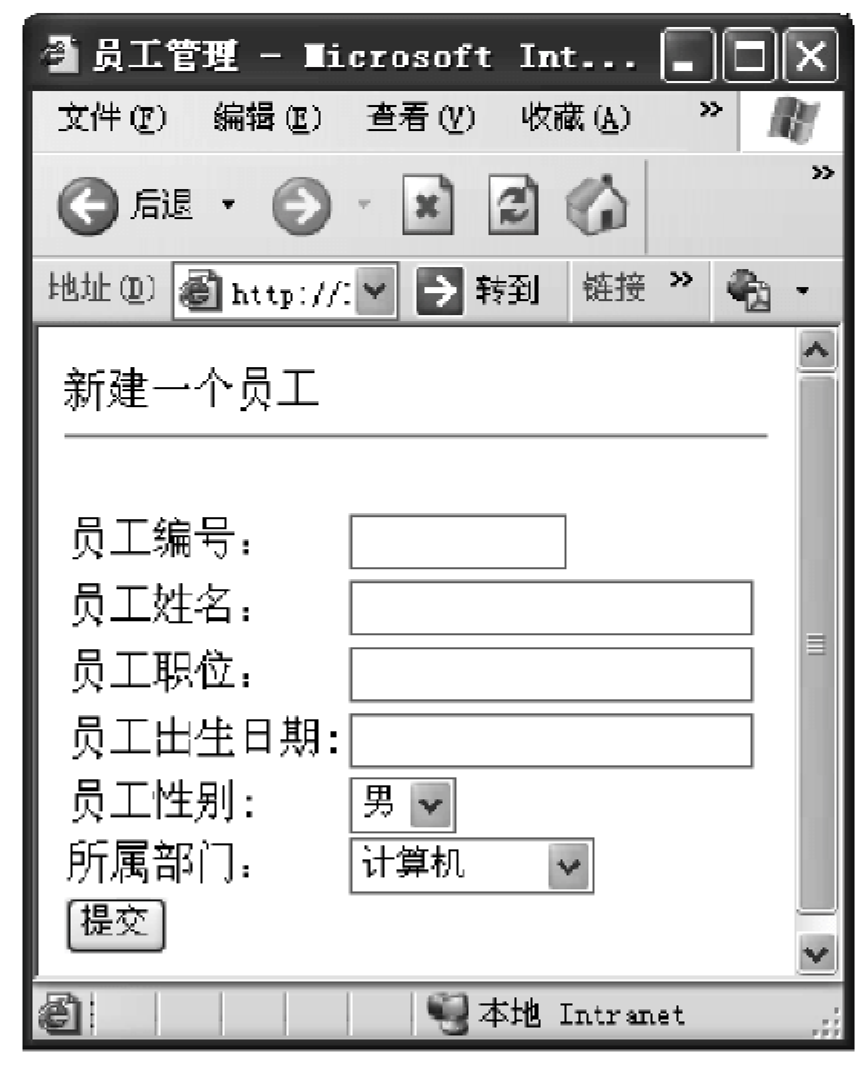
程序清单 11-10:

```
<%--employeeedit.jsp--%>
<%
  代码段 2;
  //修改记录
  response.sendRedirect("employeeRW.jsp");
%>
```

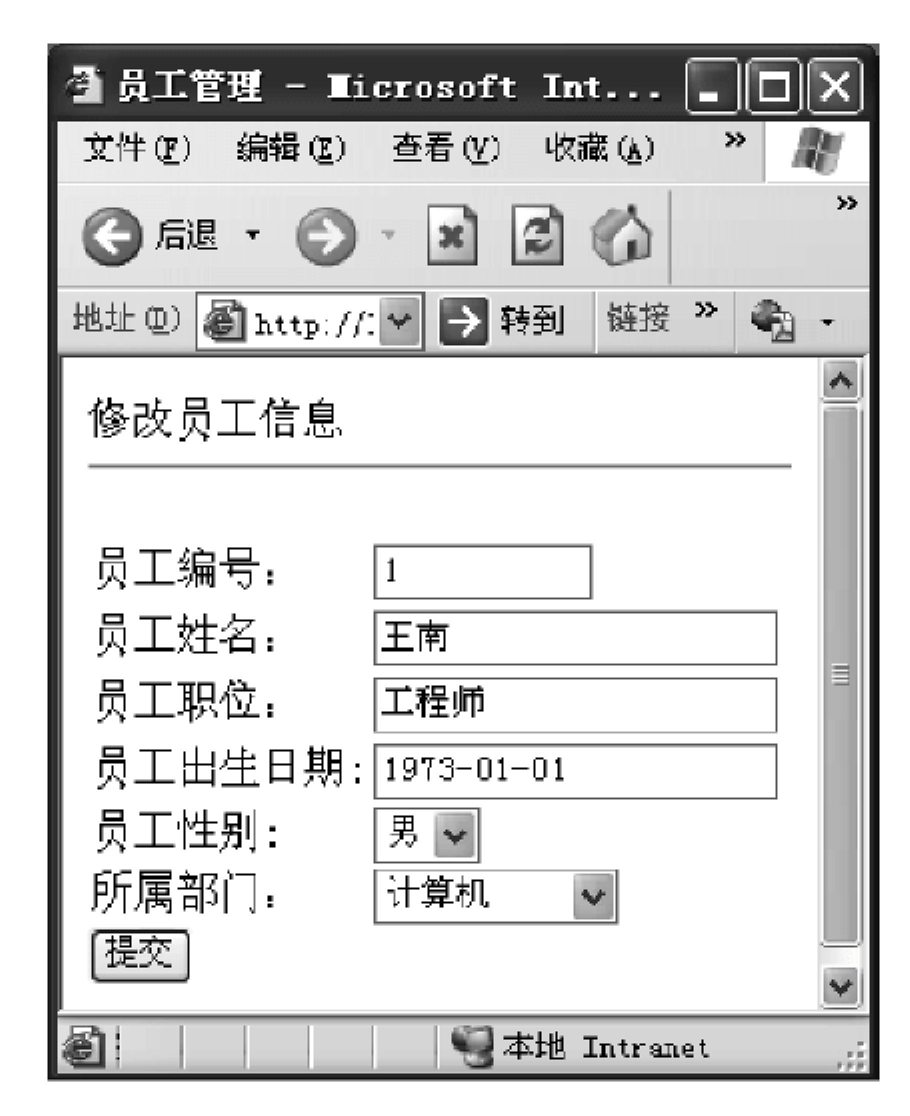
程序清单 11-11:

```
<%--employeedelete.jsp--%>
<jsp:useBean id="employee" class="chapter11.Employee" scope="page"/>
<%
  String id=request.getParameter("id");
  代码段 3;
  //删除指定编号的记录
  response.sendRedirect("employeeRW.jsp");
%>
```

(5) 保存以上程序,在 Tomcat 服务器中运行和测试,保证程序能够正确地处理数据库中的数据。运行效果如图 11-10 所示。



(a) 新建一个员工



(b) 修改员工信息

图 11-10 员工信息更新

11.5　实验 11.4　导出数据库数据到本地文件

实验目的：

（1）掌握 JDBC 的基本结构和主要接口使用方法。
（2）学会在 JSP 中数据库操作。

实验内容：

本次实验训练读者掌握 JSP 访问数据库的使用技术。在实际的开发中往往需要将一些数据整理成报表，而 Excel 是 Office 家族中常用的电子表格软件，故在 JSP 开发中经常会把数据库中的数据导出到本地的 Excel 文件。本次实验编写数据 Excel 导出程序。

实验步骤：

（1）修改实验 11.3 中程序清单 11-4 的 employeeRW.jsp，在代码中加入一个链接用来启动 Excel 文件导出的 JSP 代码，代码如下所示：

```
<a href="download.jsp">下载数据</a>
```

（2）新建 download.jsp，将普通的数据文件在网页中显示，修改自动启动文件下载到本地。将程序清单 11-12 代码输入。

程序清单 11-12

```
<!--download.jsp-->
<%@page contentType="application/msexcel; charset=gb2312" language="java"%>
<%@age import="java.sql.*"%>
<html><head>
    <title>员工表</title>
  </head>
  <%
   try{
    Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver").newInstance();
    String url="jdbc:sqlserver://localhost:1433;DatabaseName=MySampleTest";
    Connection conn=DriverManager.getConnection(url,"user1","user1");
    Statement stmt=conn.createStatement();
  %>
  <body>
    <table border="1" cellspacing="1">
      <tr>
        <td>员工编号</td>
        <td>员工姓名</td>
        <td>员工职位</td>
        <td>员工出生日期</td>
        <td>员工性别</td>
        <td>部门名称</td>
```

```
        </tr>
        <%
          String sql="select 员工编号,员工姓名,员工职位,员工出生日期,员工性别,部门名称 ";
          sql=sql+"from 员工 left outer join 部门 on (员工.所属部门编号=部门.部门编号)";
          ResultSet rs=stmt.executeQuery(sql);
          while(rs!=null && rs.next()){
            out.print("<tr><td>"+rs.getInt(1)+"</td>");
            out.print("<td>"+rs.getString(2)+"</td>");
            out.print("<td>"+rs.getString(3)+"</td>");
            out.print("<td>"+rs.getDate(4)+"</td>");
            out.print("<td>"+rs.getString(5)+"</td>");
            out.print("<td>"+rs.getString(6)+"</td></tr>");
         }
       %>
      </table>
    </body>
    <%
      rs.close();
      stmt.close();
      conn.close();
    }catch(ClassNotFoundException cnfe){
      out.print(cnfe);
    }catch(SQLException sqle){
      out.print(sqle);
    }catch(Exception e){
      out.print(e);
    }
  %>
  <%
    response.setHeader("Content-disposition","attachment; filename=data.xls");
  %>
  </html>
```

(3) 保存文件,启动服务器测试,得到如图 11-11 运行效果。

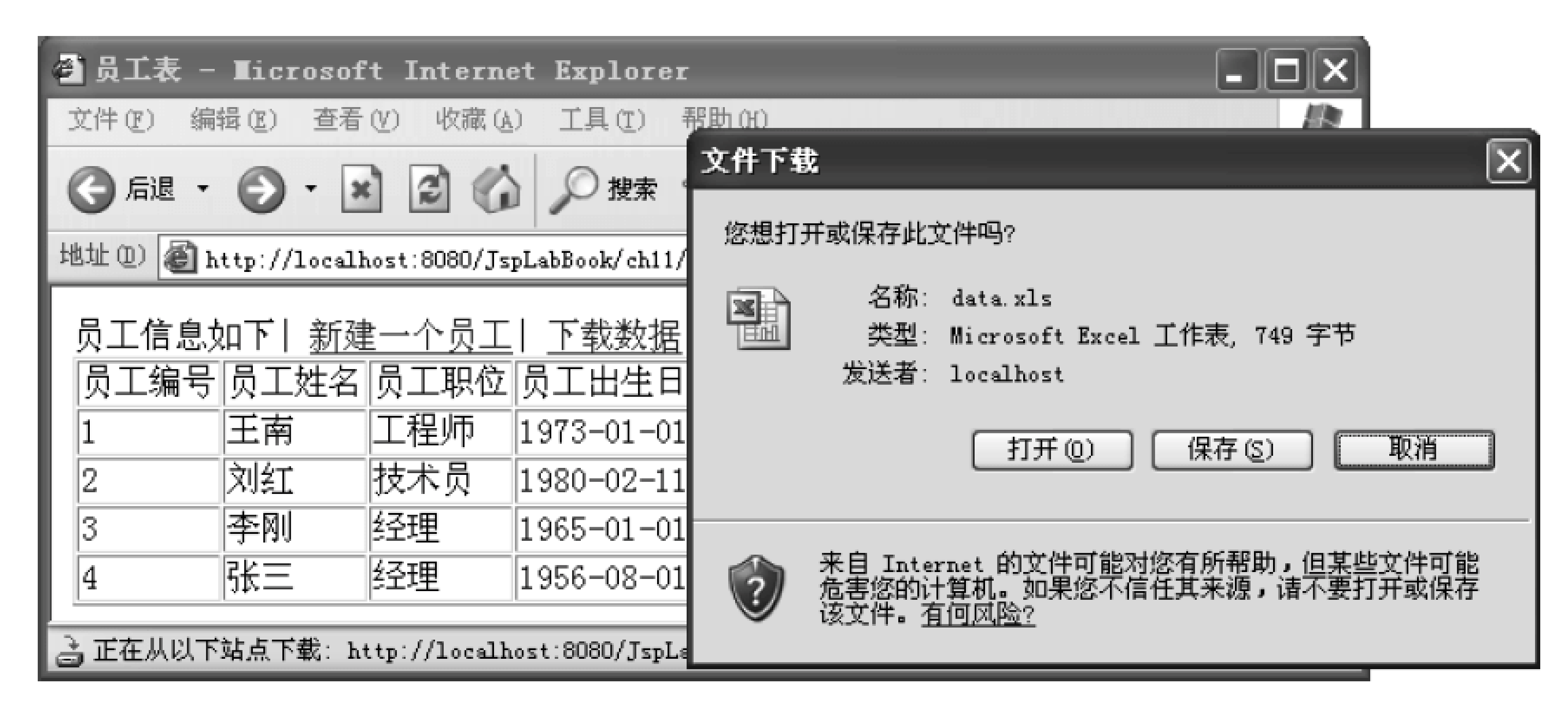


图 11-11　数据导出 Excel 文件

第 12 章　JSP 的 JavaBean 编程

JavaBean 是 Sun Microsystems 公司于 1997 年发布的一种可重用的组件。JavaBean 往往与 JSP 结合使用,已经成为 JSP 两大 MVC 开发模式之一。JavaBean 的出现,进一步实现 Java 代码与网页的数据的分离,提高了 JSP 文件的可维护性,降低了开发人员对应用 Java 语言的要求。为此,本章简略介绍了 JavaBean 编程的要点以及设计与 JavaBean 编程相关的实验。

12.1　预 备 知 识

JavaBean 用 Java 语言编写的可重用的软件组件。这种软件组件实质上就是 Java 公共类,只是这种公共类特殊在于只有一个不带参数的默认构造方法,并预先定义了特定的属性和方法,通常有 setXXX()和 getXXX()方法实现对特定属性的设置和获取。JavaBean 不但编写简单,容易学习,而且可根据用户的需要实现多种复杂的业务逻辑。从这个角度上来说,有必要对 JavaBean 做一个了解。下面将从 JavaBean 的属性、JavaBean 的应用以及 JavaBean 的作用域 3 个方面,对 JavaBean 做一个介绍。

12.1.1　JavaBean 的属性

JavaBean 的属性体现了 JavaBean 的内部特征,是 JavaBean 内部状况的抽象表示。依据应用中的不同要求,以及 Sun Systems 公司提供的 JavaBean 技术规范,可以将 JavaBean 的属性分成 4 种类型:简单类型、索引类型、绑定类型和约束类型,具体内容见表 12-1。

表 12-1　JavaBean 的属性

类　型	形　　式	说　　明

简单类型	void set<Property> (PropertyType value)
	//设置属性
	PropertyType get<Property> ()
	//获取属性
	或
	void set<Property> (boolean value)
	//设置布尔类型的属性
	boolean is<Property> ()　　//获取判断
		针对单值属性而言,是通过 JavaBean 的相关方法的调用实现对 JavaBean 属性的访问

索引类型	void set<Property> (int index,<PropertyType>value);
	//索引设置
	void set<Property> (<PropertyType[]>value);
	//设置属性的一组值
	void PropertyType get<Property> (int index);
	//索引获取;
	void PropertyType[] get<Property> ();
	//获取属性的一组值
		针对属性对应指定范围的一组值。要访问属性中一组值的一个必须通过索引实现

续表

类　型	形　　式	说　　明

绑定类型	public
	void
	addPropertyChangeListener(PropertyChangeListener x);
	//增加属性变化监听者
	public
	void
	removePropertyChangeListener(PropertyChangeListener x);
	//移除属性变化监听者
		在 JavaBean 对象的属性发生变化，就要发送一个 PropertyChange 事件通知所有相关的监视器，然后进行处理

约束类型	PropertyType get<Property>();　　//获取属性
	void set<Property>(PropertyType value) throws
	PropertyVetoException;
	//设置属性,属性改变抛出约束属性异常
	public
	void
	addVetoableChangeListener(VetoableChangeListener x);
	//增加约束监听者
	public
	void
	removeVetoableChangeListener(VetoableChangeListener x);
	//移除约束监听者
		表示 JavaBean 的属性受到约束。这种约束由监听器来决定是否要求否决属性发生的任何一个变化，强迫返回原来的设置

一般，JSP 多使用具有简单类型和索引属性的 JavaBean 组件。约束类型和绑定类型多用在 GUI 图形组件中。

12.1.2　JavaBean 的访问

定义一个 JavaBean 组件的最终目的是应用它。要实现访问 JavaBean 组件需要在 JSP 文件中依照下列步骤来实现。

1. 导入 JavaBean 的类

在 JSP 文件中，通过指令 page 来导入 JavaBean 的类。具体形式如下：

```
<%@page import="类名"%>
```

2. 创建一个 JavaBean 实例对象

导入成功后，在 JSP 文件中通过<jsp:useBean>创建一个指定作用域的 JavaBean 的实例对象。具体形式如下：

```
<jsp:useBean id="对象名" scope="作用域名" class="类名"| type="类型名"|beanName="bean 的
名字">
⋮
</jsp:useBean>
```

其中：

① id：定义对象实例名。

② scope：说明 JavaBean 的作用域。

③ class：说明 JavaBean 的类名，可以单独使用，也可以和 type 结合使用。

④ type：说明 JavaBean 的类型，可以单独使用，也可以与 class 或 beanName 结合使用。

⑤ beanName：说明 bean 的名字，只能与 type 结合使用。

3. 使用 JavaBean 实例对象的属性

使用 JavaBean 实例对象的属性有两种情况：设置属性和获取属性值。JSP 为此分别提供了<jsp:setProperty>动作和<jsp:getProperty>动作来实现属性的应用。

(1) <jsp:setProperty>。

<jsp:setProperty>动作为指定 JavaBean 对象设置属性值，表示形式如下：

```
<jsp:setProperty name="bean的名字" property="* |属性名"
                                [param="参数名"]
                                [value="值"]
>
```

其中：

① name：必选项，指定 JavaBean 实例对象的名字。

② property：必选项，说明设置的属性。如果为 *，表示设置或修改所有的属性；如果为属性名，则设置或修改指定属性。

③ param：可选项，表示用户请求的参数，与 Web 表单内容一致。它必须与 property 结合使用。

④ value：可选项，表示属性值。取值可以是字符串也可以是表达式形式。value 必须与 property 结合使用。

(2) <jsp:getProperty>。

<jsp:getProperty>动作可以获取 JavaBean 实例对象的属性值，形式如下：

```
<jsp:getProperty name="bean的名字" property="属性">
```

其中：

① name：指定 JavaBean 实例对象的名字。

② property：指定要获取对象属性的名字。

12.1.3 JavaBean 的作用域

在动作<jsp:useBean>中可以用 scope 属性设置 JavaBean 组件的作用域，指定 JavaBean 对象的应用范围。JavaBean 具有 4 种作用域：page、request、session 和 application，具体说明见表 12-2。

表 12-2 JavaBean 的作用域

作 用 域	说　明
page	表示创建的 Bean 对象只在当前 JSP 页面有效
request	表示创建的 Bean 对象只在本次请求有效，具体作用在当前 JSP 页面，以及 JSP 页面中用<%@ include%>指令包含的页面以及用<jsp:forward>动作转向的页面
session	表示在用户与服务器交互的会话过程中有效
application	表示在整个 Web 应用中有效

12.2　实验 12.1　JSP 使用 JavaBean

实验目的：

（1）了解 JavaBean 的基本概念。

（2）了解和熟练掌握 JavaBean 开发的基本步骤和关键语法。

（3）了解和熟练掌握 JavaBean 的作用域的不同作用范围，并结合实际应用。

（4）熟练和灵活运用 JavaBean 属性类型的不同，来解决实际问题。

实验内容：

设计一个用 Windows Media Player 播放 MP3 音乐的程序。为《River Dance》音乐专辑的 18 首 MP3 音乐设计播放程序。利用 JavaBean 来保存用户选择的音乐。该实验由 3 个练习构成。

（1）利用 JavaBean 的简单属性类别，实现用户选择的单个音乐文件的播放。

（2）修改 JavaBean 的作用域，比较在不同作用域的应用。

（3）利用 JavaBean 的索引属性，实现用户选择的多个音乐文件连续播放。

实验步骤：

本次实验是帮助读者了解 JavaBean 的属性的简单类型和索引类型的应用。本次实验的内容是设计一个音乐播放程序，实现用户可以用 Windows Media Player 播放《River Dance》音乐专辑的 18 首 MP3 音乐文件。

练习 1：用简单属性 JavaBean 实现单首音乐文件播放。

已提供 3 个文件 MusicBean.java、music.jsp 和 playMusic.jsp，对应的代码分别见程序清单 12-1～12-3。其中，MusicBean.java 是保存用户选择的音乐文件名的 JavaBean 程序；music.jsp 是显示《River Dance》音乐专辑的 18 首 MP3 音乐列表，让用户根据列表进行选择；playMusic.jsp 是提供实现播放用户选择的单首音乐文件的部分代码。请修改程序清单 12-3，使之运行结果能如图 12-1 和图 12-2 所示。

程序清单 12-1：

```
//MusicBean.java
package beans;
public class MusicBean {
    private String name;
    public String getName() {                          //获取音乐文件名
        return name+".mp3";
    }
    public String getTitle(){                          //获取音乐标题
        int i;
        String title="";
        try{
            i=Integer.parseInt(name);
```

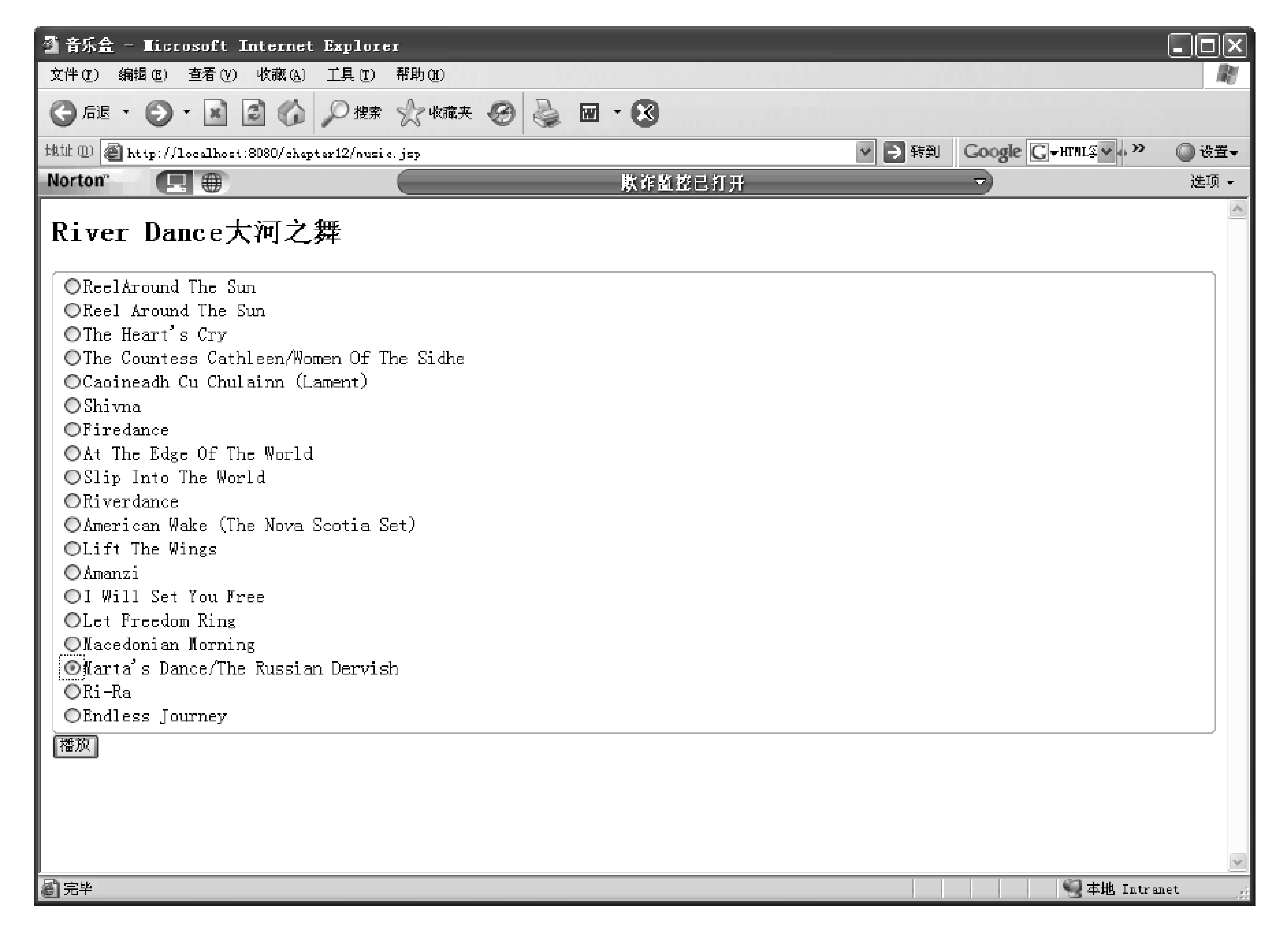


图 12-1　显示音乐列表

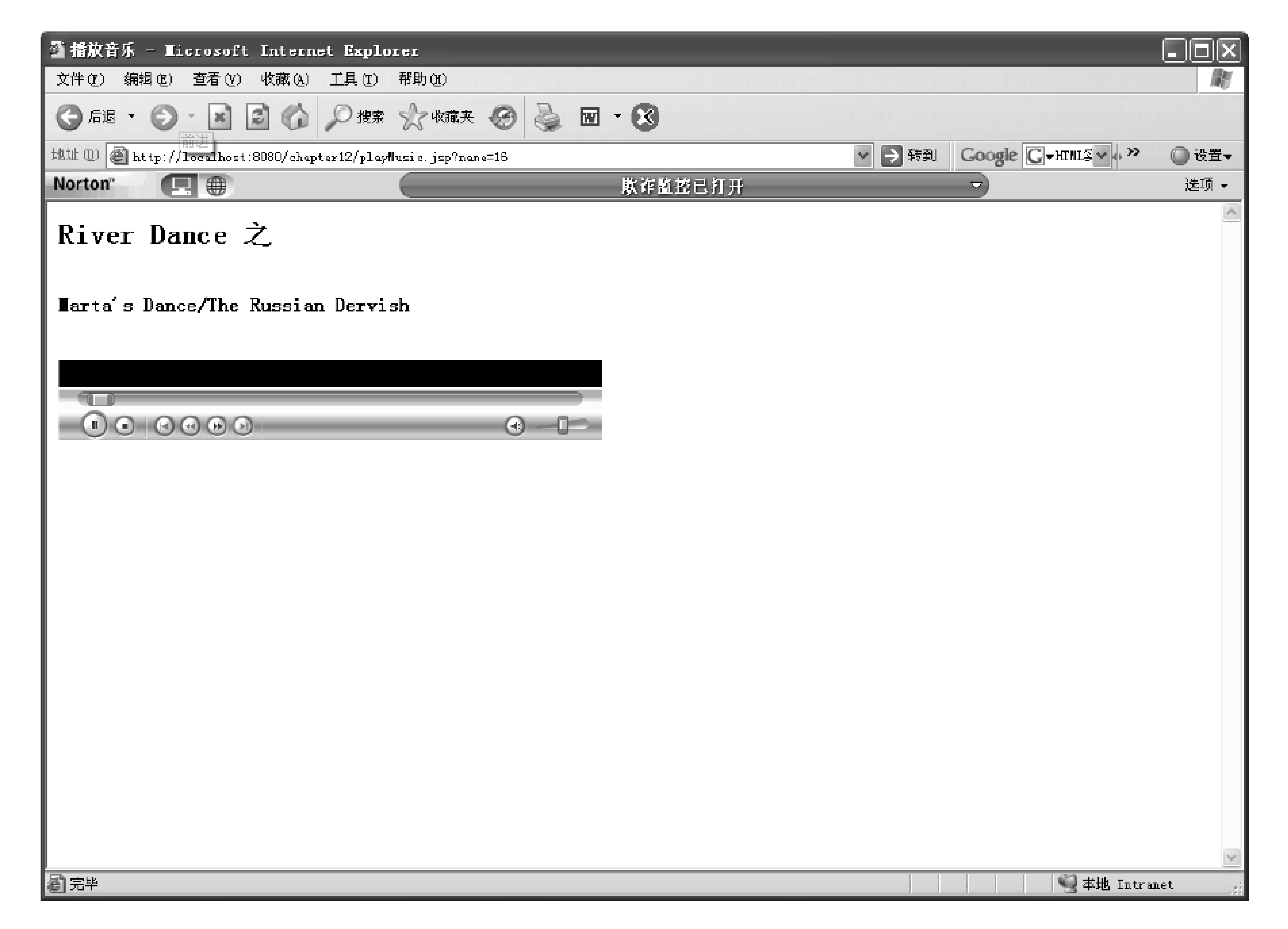


图 12-2　播放音乐

```
}catch(Exception e){
    i=1;
}
switch(i){
case 1:title="Reel Around The Sun";break;
case 2:title="The Heart's Cry";break;
case 3:title="The Countess Cathleen/Women Of The Sidhe";break;
```

```
        case 4:title="Caoineadh Cu Chulainn (Lament)";break;
        case 5:title="Shivna";break;
        case 6:title="Firedance";break;
        case 7:title="At The Edge Of The World";break;
        case 8:title="Slip Into The World";break;
        case 9:title="Riverdance";break;
        case 10:title="American Wake (The Nova Scotia Set)";break;
        case 11:title="Lift The Wing";break;
        case 12:title="Amanzi";break;
        case 13:title="I Will Set You Free";break;
        case 14:title="Let Freedom Ring";break;
        case 15:title="Macedonian Morning";break;
        case 16:title="Marta's Dance/The Russian Dervish";break;
        case 17:title="Ri-Ra";break;
        case 18:title="Endless Journey";break;
        }
        return title;
    }
    public void setName(String name) {                //设置音乐名
        this.name=name;
    }
}
```

程序清单 12-2:

```
<!--music.jsp-->
<%@page contentType="text/html; charset=gb2312" language="java"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type"content="text/html; charset=gb2312" />
<title>音乐盒</title>
</head>

<body>
<p><h2>River Dance 大河之舞</h2></p>
<form method="get" action="playMusic.jsp">
  <fieldset title="曲目">
    <input type="radio" name="name" value="1">
      ReelAround The Sun
    </input><br  />
    <input type="radio" name="name" value="2">The Heart's Cry</input><br  />
    <input type="radio" name="name" value="3">
      The Countess Cathleen/Women Of The Sidhe
    </input>
```

```
<br />
<input type="radio" name="name" value="4">
  Caoineadh Cu Chulainn (Lament)
</input>
<br />
<input type="radio" name="name" value="5">
  Shivna
</input>
<br />
<input type="radio" name="name" value="6">
  Firedance
</input>
<br />
<input type="radio" name="name" value="7">
  At The Edge Of The World
</input>
<br />
<input type="radio" name="name" value="8">
  Slip Into The World
</input>
<br />
<input type="radio" name="name" value="9">
  Riverdance
</input>
<br />
<input type="radio" name="name" value="10">
  American Wake (The Nova Scotia Set)
</input>
<br />
<input type="radio" name="name" value="11">
  Lift The Wings
</input>
<br />
<input type="radio" name="name" value="12">
  Amanzi
</input>
<br />
<input type="radio" name="name" value="13">
  I Will Set You Free
</input>
  <br />
<input type="radio" name="name" value="14">
  Let Freedom Ring
</input>
<br />
```

```
<input type="radio" name="name" value="15">
Macedonian Morning
</input>
<br  />
<input type="radio" name="name" value="16">
  Marta's Dance/The Russian Dervish
</input>
<br  />
<input type="radio" name="name" value="17">
  Ri-Ra
</input>
<br  />
<input type="radio" name="name" value="18">
Endless Journey
</input>
<br  />
</fieldset>
<input type="submit" value="播放"/>
</form>
</body>
</html>
```

程序清单 12-3：

```
<!--playMusic.jsp-->
<%@page contentType="text/html; charset=gb2312"  language="java" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>播放音乐</title>
</head>

<body>
<jsp:useBean id="musicBean" scope="page" class="beans.MusicBean"/>
<%
if(musicBean.getName()!=null){
out.println("<embed src=media/"+musicBean.getName()+" loop=false autostart=true name=bgm
width='460' height='68'/>");
}
%>
</body>
</html>
```

问题： 如果将程序清单 12-3 中的倒数第 4 行代码中的 musicBean. getName( )

用<jsp:getProperty>代替，该如何修改该程序。

练习 2：了解 JavaBean 的作用域。

已知有一个 MusicHistoryBean.java，代码见程序清单 12-4，可用来记录播放乐曲的历史记录。修改上述的 playMusic.jsp，代码见程序清单 12-5，使之可以显示已播放的历史记录。将 playMusic.jsp 的第 17 行语句"<jsp:useBean id="historyBean" scope="page" class="beans.MusicHistoryBean" />"中的 scope 属性分别用 page、request、session 和 application 替代，打开不同浏览器窗口，依次运行不同作用域的程序，观察运行结果，并解释运行结果不同的原因。

程序清单 12-4：

```
//MusicHistoryBean.java
package beans;

public class MusicHistoryBean {
  private String history;                          //音乐历史记录
  public void setHistory(String history){
      this.history=history;
  }
  public String getHistory(){
      return this.history;
  }
}
```

程序清单 12-5：

```
<!--修改后的 playMusic.jsp-->
<%@page contentType="text/html; charset=gb2312"
         language="java"
         import="beans.*"
%>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/
xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>播放音乐</title>
</head>

<body>
<jsp:useBean id="musicBean" scope="page" class="beans.MusicBean">
<jsp:setProperty name="musicBean" property="*"/>
</jsp:useBean>
<jsp:useBean id="historyBean" scope="session" class="beans.MusicHistoryBean"/>
<%
String title=musicBean.getTitle();                        //正在播放音乐的标题
```

```
if(title!=null){
  out.println("<h2>River Dance 之</h2><br/><h4>"+title+"</h4><br/>");
  String historylist=historyBean.getHistory();            //获取已播放音乐的历史记录
   if(historylist==null) historylist="";
   historylist+=musicBean.getTitle()+"<br/>";             //增加新的记录
%>
   <jsp:setProperty name="historyBean" property="history" value="<%=historylist%>"/>
                                                          <!--重新设置历史记录-->
<%
   if(musicBean.getName()!=null)
   out.println("<embed src=media/"+musicBean.getName()+" loop=false autostart=true name
=bgss width='460'height='68'/>");
   }
%>
<hr/>
<p>
<h3>已经播放的乐曲名称:</h3><br/>
<jsp:getProperty name="historyBean" property="history"/>
</p>
</body>
</html>
```



图 12-3　播放音乐的历史记录

练习 3：用索引属性的 JavaBean 实现多个音乐文件连续播放。

已知 3 个程序 MusicListBean.java、musicList.jsp 和 playMusic2.jsp，通过它们实现用户可以选择多个音乐的 MP3 文件，而且这些文件可以连续播放。其中，musicList.jsp 是显示音乐列表的多选表单，对应的代码见程序清单 12-6；playMusic2.jsp 是显示选中音乐的名称以及连续播放这些音乐，对应的代码见程序清单 12-7。MusicListBean.java 定义设置和

获取选中音乐文件名的 JavaBean，并可以创建播放列表。请仔细阅读和理解程序，将程序清单 12-8 中的 4 个方法 setList(String[] value)、getName(int index)、getList()和 getNameList()补充完整，完整定义 MusicListBean.java。然后，编译和调试程序，使运行结果如图 12-4 和图 12-5 所示。

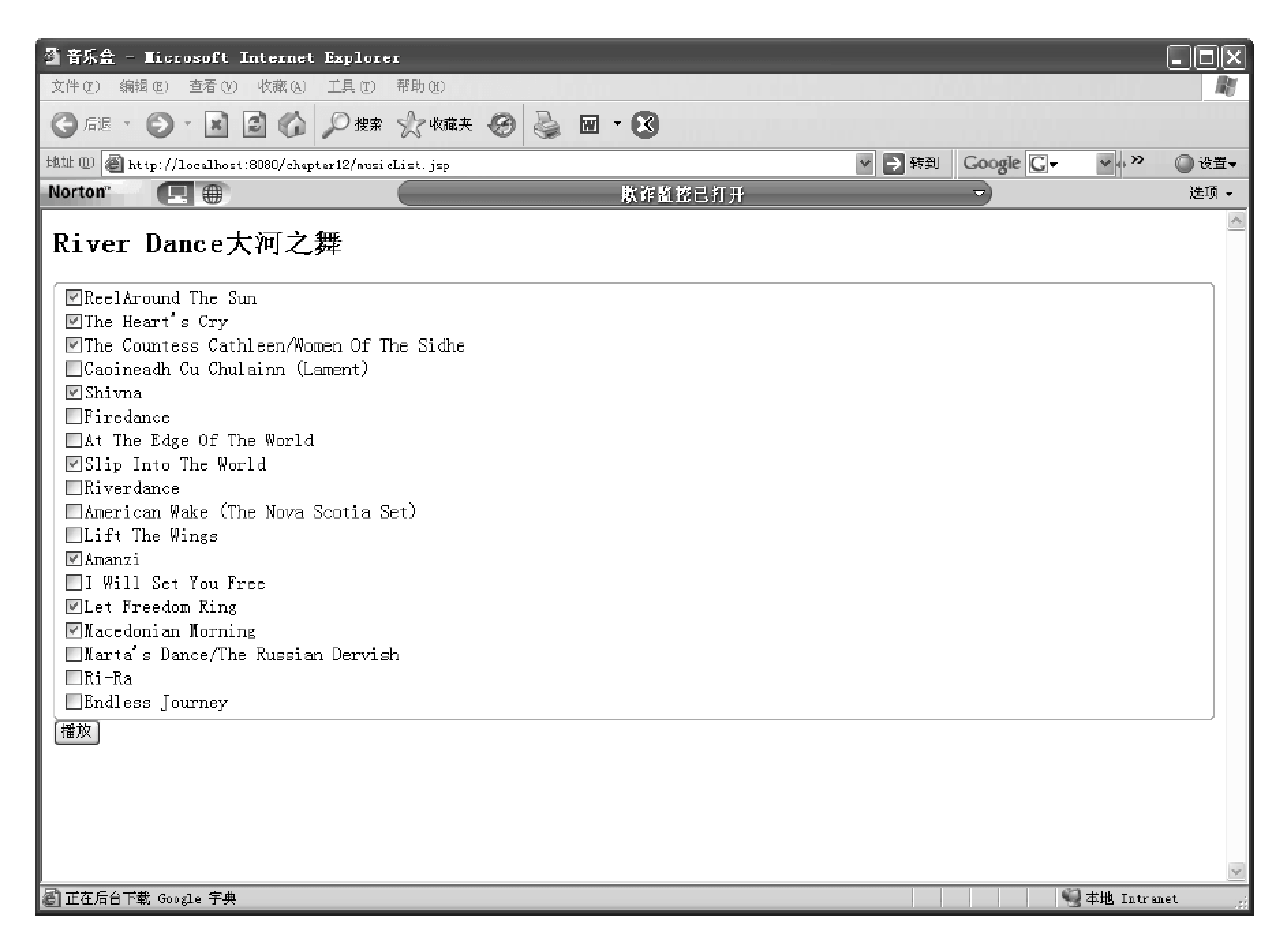


图 12-4　显示多选列表

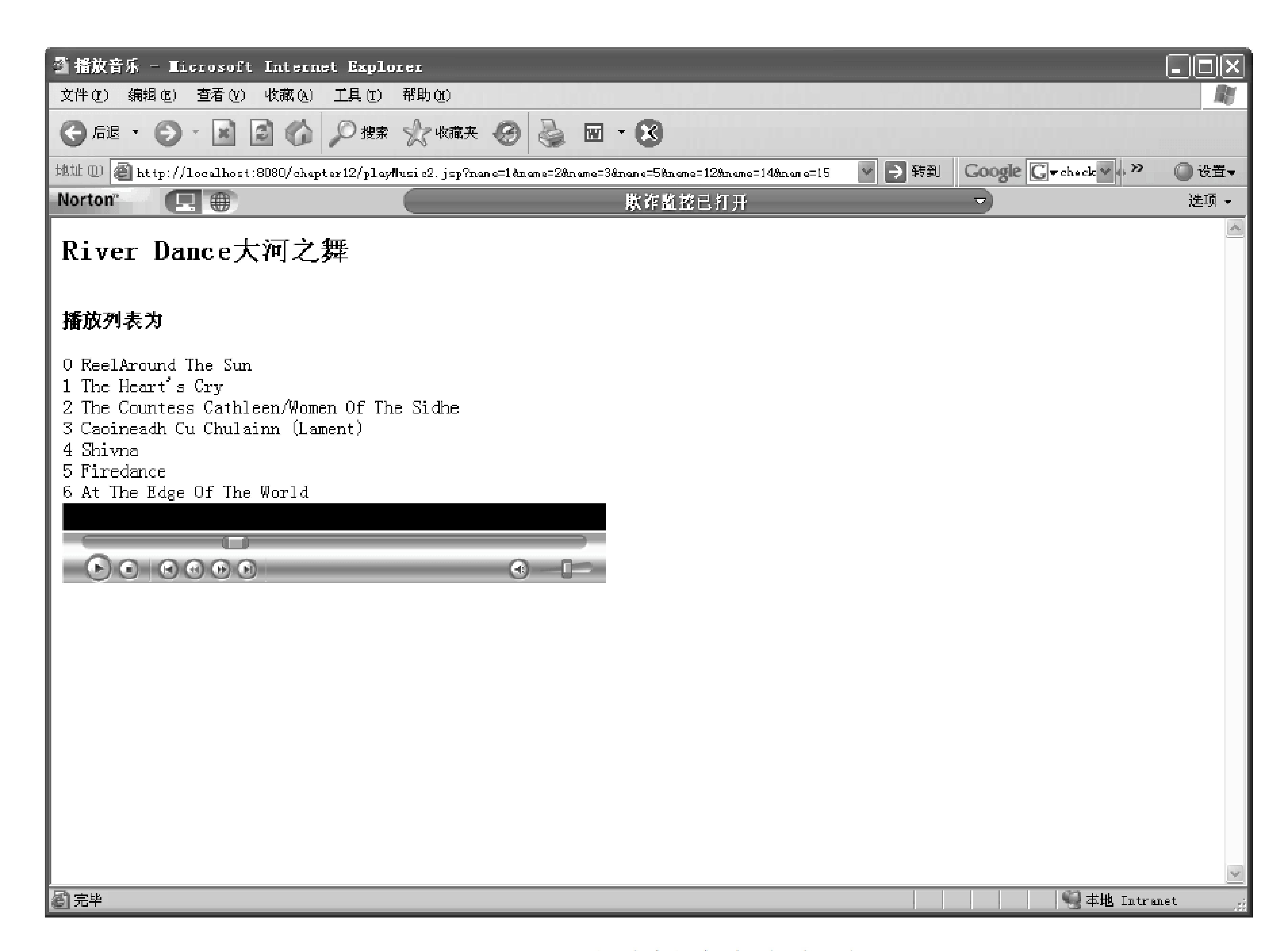


图 12-5　连续播放多个音乐

程序清单 12-6：

```
<!--musicList.jsp-->
<%@page contentType="text/html; charset=gb2312"
        language="java"
        %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
                    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>音乐盒</title>
</head>

<body>
<p><h2>River Dance 大河之舞</h2></p>
<form name="testForm" method="get" action="playMusic2.jsp">
  <fieldset title="曲目">
    <input type="checkbox" name="name" value="1">
      ReelAround The Sun
    </input>
    <br/>
    <input type="checkbox" name="name" value="2">
      The Heart's Cry
    </input><br/>
    <input type="checkbox" name="name" value="3">
      The Countess Cathleen/Women Of The Sidhe
    </input>
    <br/>
    <input type="checkbox" name="name" value="4">
      Caoineadh Cu Chulainn (Lament)
    </input>
    <br/>
    <input type="checkbox" name="name" value="5">
      Shivna
    </input>
    <br/>
    <input type="checkbox" name="name" value="6">
      Firedance
    </input>
    <br/>
    <input type="checkbox" name="name" value="7">
      At The Edge Of The World
    </input>
    <br/>
    <input type="checkbox" name="name" value="8">
      Slip Into The World
```

```
        </input>
        <br/>
        <input type="checkbox" name="name" value="9">
          Riverdance
        </input>
        <br/>
        <input type="checkbox" name="name" value="10">
          American Wake (The Nova Scotia Set)
        </input>
        <br/>
        <input type="checkbox" name="name" value="11">
          Lift The Wings
        </input>
        <br/>
        <input type="checkbox" name="name" value="12">
          Amanzi
        </input>
          <br/>
        <input type="checkbox" name="name" value="13">
          I Will Set You Free
        </input>
        <br/>
        <input type="checkbox" name="name" value="14">
          Let Freedom Ring
        </input>
        <br/>
        <input type="checkbox" name="name" value="15">
          Macedonian Morning
        </input>
        <br/>
        <input type="checkbox" name="name" value="16">
          Marta's Dance/The Russian Dervish
        </input>
        <br/>
        <input type="checkbox" name="name" value="17">
          Ri-Ra
        </input>
        <br/>
        <input type="checkbox" name="name" value="18">
          Endless Journey
        </input>
        <br/>
    </fieldset>
    <input type="submit" value="播放"/>
</form>
```

```
</body>
</html>
```

程序清单 12-7：

```
<!--playMusic2.jsp-->
<%@page contentType="text/html; charset=gb2312"    language="java"
          import="beans.MusicListBean"
%>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>播放音乐</title>
</head>

<body>
<jsp:useBean id="listBean" scope="page" class="beans.MusicListBean"/>
<p>
<h2>River Dance 大河之舞</h2><br/>
<h4>播放列表为</h4>
<%
String[] mlist=request.getParameterValues("name");     //获取复选框选中的值
if(mlist!=null)                                        //用户选中至少一项
listBean.setList(mlist);                               //设置选中的索引号
int[] index=listBean.getList();                        //获取整型的索引号
for(int i=0;i<index.length;i++)                        //输出编号和选中的音乐名
  out.println(""+i+" "+listBean.getName(i)+"<br/>");
listBean.createASXFile(request);                        //创建临时视频播放列表 asx 文件
out.println("<embed src=media/musiclist.asx"+
                 "loop=false autostart=true name=bgss width='460' height='68'/>");
%>
</p>
</body>
</html>
```

程序清单 12-8：

```
//MusicListBean.java
package beans;
import java.io.*;
import javax.servlet.http.*;
public class MusicListBean {
  private String[] names={                              //设置音乐名;
                          "ReelAround The Sun",
```

```
                            "The Heart's Cry",
                            …                               //中间略
                            "Endless Journey"};
private int[] list;                                         //选中音乐列表名索引
public void setName(int index,String name){                 //设置指定位置的音乐名称
    if(index<names.length&&index>=0)
        names[index]=name;
}
public void setNameList(String[] names){                    //设置所有有效的音乐名
    this.names=names;
}
public void setList(String[] value){                        //设置选中的音乐名列表
}
public String getName(int index){                           //获取指定音乐的名称
}
public int[] getList(){                                     //获取索引列表
}
public String[] getNameList(){                              //获取所有有效的音乐名
}
public boolean createASXFile(HttpServletRequest request){             //创建播放列表文件
    String path=request.getRealPath(".");
    try{
    File f=new File(path+"\\media\\musiclist.asx");
    if(f.exists())
         f.delete();                                                  //如果存在删除
    FileWriter fw=new FileWriter(path+"\\media\\musiclist.asx");
    BufferedWriter bw=new BufferedWriter(fw);
    bw.write("<asx version=\"3.0\">");
    bw.newLine();
    if(list==null) return false;
    for(int i=0;i<list.length;i++){
      bw.write("<entry>");
      bw.newLine();
      bw.write("<ref href=\"http://localhost:8080/chapter12/media/"+(list[i]+1)+".mp3
      \"/>");
      bw.newLine();
      bw.write("<title>"+names[list[i]]+"</title>");
      bw.newLine();
      bw.write("</entry>");
      bw.newLine();
    }
    bw.write("</asx>");
    bw.flush();                                                       //将数据更新至文件
    fw.close();                                                       //关闭文件流
    }catch(IOException e){
```

```
            return false;
        }
        return true;
    }
}
```

说明：asx 文件是用 XML 表示播放信息的播放列表文件。一般形式如下：

```
<asx version="3.0">
  <entry>                                  <!--播放文件信息-->
    <ref>…</ref>                           <!--说明播放文件位置-->
    <title>….</title>                      <!--说明播放音乐的名称-->
    <author>…</author>                     <!--作者-->
     ⋮
   </entry>
   ⋮
   <entry>
       ⋮
   </entry>
</asx>
```

12.3 实验 12.2 JavaBean 连接数据库

实验目的：

(1) 进一步了解 JavaBean 的应用。
(2) 了解 JavaBean 实现数据库的访问。
(3) 初步了解 JSP＋JavaBean 的工作模式。
(4) 运用 JavaBean 访问数据库，开发具有实际意义的应用。
(5) 深入了解和应用 JDBC 访问 Web 数据库。

实验内容：

ForumLite 2.1 是 Internet 上用 PHP 语言开发的一个最小论坛系统。模仿 ForumLite 2.1 的结构设计一个具有最基本功能的简易论坛。功能要求如下。

此系统要实现允许用户浏览论坛、发表新论点、也可以浏览并回复论坛的已有论点。论坛中所有的数据信息用 MySQL 数据库保存。要求用 JavaBean 处理对数据库的访问。具体功能要求。

(1) 要求实现用户的注册与登录。

(2) 任何用户可以直接发表论题，论题包括标题、内容、发表者的电子邮件和发表日期；如果有用户对该论题回复，要求能显示已回复论题的次数。

(3) 任何用户可以浏览已发表的论题，并可以针对特定的论题进行回复。回复信息包括标题、内容、回复者的电子邮件以及回复的时间。

(4) 要求所有论题以分页的形式显示,每页面 10 个论题。

(5) 要求某论题的所有回复也以分页显示,每页面显示 10 个回复。

实验步骤:

本次实验主要有两个具体任务:利用 JavaBean 访问 MySQL 数据库,实现对保存论坛信息的 MySQL 的连接,以及对论坛的浏览和论题回复信息的添加;利用 JavaBean 实现数据的分页访问。要求如下。

(1) 用 MySQL 数据库保存信息。请用 MySQL 数据库,建立一个数据库 ForumDB,在数据库 ForumDB 下建立一个保存论坛论题的数据表 ForumsTable 和保存回复信息的数据表 ReplysTable。建立数据库和数据表的文件见程序清单 12-9 的 ForumDB. sql。

程序清单 12-9:

```
/* ForumDB.sql */
create database if not exists ForumDB;                      //创建数据库 ForumDB
use ForumDB;                                                //打开数据库 ForumDB
create table if not exists ForumsTable(                     //创建 ForumsTable 表,保存论坛
                                                            //主题信息
    forum_id INT UNSIGNED AUTO_INCREMENT NOT NULL,          //定义主题编号,自动编码类型非空
    PRIMARY KEY(forum_id),                                  //forum_id 为主键
    ftitle VARCHAR(50) NOT NULL,                            //标题非空
    ftime   INT   NOT NULL,                                 //回复主题的次数非空
    fauthor VARCHAR(20) NOT NULL,                           //作者名
    femail VARCHAR(50) NOT NULL,                            //发论题的 email
    fdate DATETIME NOT NULL,                                //创建论题的时间
    fcontent TEXT                                           //论题的内容
);
create table if not exists ReplysTable(                     //创建 ReplysTable 表,保存回复
    reply_id INT UNSIGNED AUTO_INCREMENT NOT NULL,          //定义回复编号,自动编码非空
    PRIMARY KEY(reply_id),                                  //reply_id 为主键
    f_id INT NOT NULL,                                      //论题的编号
    rtitle VARCHAR(50) NOT NULL,                            //回复的标题
    rauthor VARCHAR(20) NOT NULL,                           //回复的作者
    remail VARCHAR(50) NOT NULL,                            //回复信息的 E-mail
    rdate DATETIME NOT NULL,                                //回复论题的时间
    rcontent TEXT                                           //回复的内容
);
```

(2) 已知有 3 个 JavaBean 程序:TopicBean. java、ConnectDBBean. java 和 PageBean. java。这 3 个程序分别实现保存论坛发表的意见、数据库的连接和对分页的处理。其中 TopicBean. java 如程序清单 12-10 所示。要求:自行设计实现其他 JavaBean,实现对两个数据表的访问,以及编写相关的 JSP 文件,使运行结果如图 12-6～图 12-9 所示。

ForumLite 2(JSP模式一:jsp+javabean)登录 - Windows Internet Explorer
文件(F) 编辑(E) 查看(V) 收藏夹(A) 工具(T) 帮助(H)
http://localhost:8080/web/courses/chapter_12/ForumLite/register.jsp
ForumLite 2(JSP模式... jsp中文问题解决方案(...

FORUM 论坛用户注册

E-Mail	chenyixxxx@163.com
昵称	陈铁
密码	••••••
确认密码	••••••
	注册

模仿ForumLite编写的最小最简单的论坛。
© 2008年8月10日《Web开发技术实用教程实验指导》

图 12-6　论坛的新用户注册



图 12-7　论坛的用户登录界面

程序清单 12-10:

```
//TopicBean.java
package JavaBean;
import java.sql.* ;
/**
 * 定义表示发表意见的 JavaBean,命名为 TopicBean,
 * 可以将发表的论题和回复论题用该 TopicBean 表示。不同之处在于
 * 回复论题不需要设置回复次数。
 * /
public class TopicBean {
```



图 12-8　论坛的论题分页显示与新论题发表界面

ForumLite 2(JSP模式一:jsp+javabean) - Windows Internet Explorer
文件(F) 编辑(E) 查看(V) 收藏夹(A) 工具(T) 帮助(H)
http://localhost:8080/web/courses/chapter_12/ForumLite/reply.jsp?id=3
FORUM 论坛
欢迎CHAN来到FORUM论坛，您所处的位置是：FORUM论坛->二手资料 ，离开请退出
chan chan@yahoo.com | 2009-05-08 23:21:57.0
关于JSP的电子书
我有WEB程序设计电子书，请与我联系。
页:
- 1 -
回复论题
标题
内容
发送
模仿ForumLite编写的最小最简单的论坛。
© 2008年8月10日《Web开发技术实用教程实验指导》

图 12-9　回复的分页显示与回复论题的界面

```
    private String title;         //标题
    private String author;        //作者
    private String email;         //电子邮件
    private Timestamp date;       //日期
    private String content;       //内容
    private int times;            //如果是论题,表示回复次数;如果是回复表示要回复论题的编码

  public String getTitle() {
       return title;
  }

  public void setTitle(String title) {
       this.title=title;
  }

  public String getAuthor() {
       return author;
  }

  public void setAuthor(String author) {
       this.author=author;
  }

  public String getEmail() {
       return email;
  }
  public void setEmail(String email) {
       this.email=email;
  }
  public Timestamp getDate() {
       return date;
  }
  public void setDate(Timestamp date) {
       this.date=date;
  }
  public String getContent() {
       return content;
  }
  public void setContent(String content) {
       this.content=content;
  }
  public int getTimes() {
       return times;
  }
  public void setTimes(int times) {
```

```
        this.times=times;
    }


    }
```

(3) 观察图 12-6～图 12-9,请修改相关程序,用 JavaScript 补充对用户提交信息的验证,例如保证输入的 email 格式类似"XXX@XXX.XXX.XXX",又如输入的标题、留言人的信息分别不能超过 50 和 20 个字符。

(4) 本简单论坛中输入中文信息会出现乱码现象,请编写一个 JavaBean 程序来解决乱码现象,并修改相关程序,来解决乱码问题。

(5) 请分析自行编写的 JavaBean 程序,请将 JavaBean 程序按照功能分类,来说明 JavaBean 的主要作用。

12.4 实验 12.3 JavaBean 组件收发 E-mail

实验目的:

(1) 进一步了解 JavaBean 的高级应用。

(2) 深入了解 JSP+JavaBean 的开发模式。

(3) 初步了解 JavaMail API 应用接口的应用。

实验内容:

JavaMail API 是平台独立的应用程序接口,用于开发 E-mail 的应用,处理邮件的发送和接收。本实验应用 JavaMail API 来开发相应的 JavaBean 组件实现 E-mail 的接收和发送的功能。

实验步骤:

开发接发 E-mail 的 Web 应用要使用 JavaMail API 1.4.1 和 JavaBeans Activation Framework(JAF)。其中 JavaMail API 1.4.1 可以到"http://www.sun.com"网站下载。如果使用的是 JDK 1.6 以下版本,同样需要到 sun 网站下载 JAF。注意:如果使用 JDK 1.6 以及以上版本,则已经包含了 JAF。JavaMail API 1.4.1 常见的类以及说明见表 12-3。

表 12-3 JavaMail API 常见的类

类	说明
javax.mail.Properties	Properties 创建一个 session 对象。用于将寻找字符串 mail.smtp.host,属性值就是发送邮件的主机
javax.mail.Session	Session 类代表 JavaMail 中的一个邮件 session
javax.mail.Transport	Transport 是用来发送信息
javax.mail.MimeMessage	Message 对象将存储实际发送的电子邮件信息
javax.mail.InternetAddress	javax.mail.InternetAddress 类。用于确定邮件地址
javax.mail.Store	Store 类实现特定邮件协议上的读、写、监视、查找等操作

续表

类	说　　明
javax. mail. Folder	Folder 类用于分级组织邮件，并提供照 Javax. mail. Message 格式访问 E-mail 的能力
javax. mail. Internet. MimeMultpart	MultPart 的子类，是保存电子邮件内容的容器。它定义了增加和删除及获得电子邮件不同部分内容的方法
javax. mail. Internet. MimeBodyPart	是 BodyPart 的子类，MimeBodyPart 代表一个 MimeMessage 对象内容的一部分。每个 MimeBodyPart 被认为有两部分：MIME 类型、匹配这个类型的内容

在执行具体练习之前，请将 JavaMail API 的 mail. jar(如果 JDK 版本低于 6. 0，还要将 JAF 的 activation. jar)保存在对应 W 应用的“WEB-INF\lib”目录中。为了初步了解 JavaMail API 的应用，本实验提供一个简易实现收发邮件的 JavaBean 程序 MailBean. java，具体代码见程序清单 12-11。

程序清单 12-11：

```
//MailBean.java
package beans.mail;

import java.io.BufferedReader;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.*;

import javax.mail.*;
import javax.mail.internet.*;

import javax.activation.*;
public class MailBean {

private String username="";                       //用户名
private String password="";                       //密码
private String from="";                           //寄信人
private String to="";                             //收信人
private String copyto="";                         //转发收信人列表
private String subject="";                        //主题
private String message="";                        //邮件内容
private String attachedFile="";                   //附件
private String smtpHost="smtp.xxx.xxx.xx";        //发送邮件的 SMTP 服务器
private String pop3Host="pop3.xxx.xxx.xx";        //收邮件的 Pop3 服务器
private MimeMessage mimeObj;                      //MIME 邮件对象

private Properties props;                         //系统属性
private boolean needAuth=false;                   //smtp 是否需要认证
private Multipart mp;
```

```
//Multipart 对象,邮件内容,标题,附件等内容均添加到其中后再生成 MimeMessage 对象
/**
 * 邮件基本信息设置
 * /
public String getAttachedFile() {
    //获取附件的名字
    return attachedFile;
}
public void setAttachedFile(String attachedFile) {
    //设置附件名
    this.attachedFile=attachedFile;
}
public String getFrom() {
    //获取寄信人
    return from;
}
public void setFrom(String from) {
    //设置寄信人
    this.from=from;
}

public String getMessage() {
    //获取邮件内容
    return message;
}

public void setMessage(String message) {
    //设置邮件内容
    this.message=message;
}
public String getPassword() {
    //获取密码
    return password;
}

public void setPassword(String password) {
    //设置密码
    this.password=password;
}

public String getSubject() {
    //获取邮件主题
    return subject;
}
public void setSubject(String subject) {
```

```
        //设置邮件主题
        this.subject=subject;
    }

    public String getTo() {
        //获取收信人
        return to;
    }

    public void setTo(String to) {
        //设置收信人
        this.to=to;
    }
    public void setCopyto(String copyto){
        //设置邮件转发
        this.copyto=copyto;
    }

    public String getCopyto(){
        //获取转发收信人列表
        return copyto;
    }
    public String getUsername() {
        //获取用户名
        return username;
    }
    public void setUsername(String username) {
        //设置用户名
        this.username=username;
    }
    /**
     * 发邮件部分
    **/
    public void setSmtpHost(String hostName) {
        //设置 SMTP 服务器
        smtpHost=hostName;
    }

    public String getSmtpHost(){
        return smtpHost;
    }

    public boolean addAttachedFile(){
        //添加附件到邮件
        try{
```

```
            BodyPart bp=new MimeBodyPart();
            FileDataSource fileds=new FileDataSource(attachedFile);
            //创建文件数据源
            bp.setDataHandler(new DataHandler(fileds));
            //设置数据处理器
            bp.setFileName(fileds.getName());
            //设置附件文件名
            mp.addBodyPart(bp);
            //添加邮件附件
            return true;
          }
          catch(Exception e){
            System.err.println("增加邮件附件："+attachedFile+"发生错误！"+e);
            return false;
          }
  }

  public boolean addFrom(){
      //添加发信人地址到邮件
      try{
            mimeObj.setFrom(new InternetAddress(from));
            //设置发信人邮件地址
            return true;
          }
          catch(Exception e){
          return false;
          }
  }

  public boolean addMessage(){
      //添加邮件内容
      try{
            BodyPart bp=new MimeBodyPart();
            //创建保存电子邮件内容的容器
             bp.setContent("<meta http-equiv=Content-Type content=text/html; charset=
gb2312>"+message,"text/html;charset=GB2312");
            //设置邮件内容
            mp.addBodyPart(bp);
            //邮件正文添加到电子邮件
            return true;
          }
          catch(Exception e){
            System.err.println("设置邮件正文时发生错误！"+e);
            return false;
          }
```

```
}

public boolean addSubject(){
    //添加邮件主题到邮件
    try{
          mimeObj.setSubject(subject);
          return true;
        }
        catch(Exception e) {
          System.err.println("设置邮件主题发生错误!");
          return false;
        }
}

public boolean addTo(){
    //添加发信人
  if(to==null)
          return false;

  try{
    mimeObj.setRecipients(javax.mail.Message.RecipientType.TO,
                                   InternetAddress.parse(to));
    //设置收信人的地址
    return true;
  }
  catch(Exception e){
      return false;
  }
}
public boolean addCopyto(){
    //添加转发列表
    if(copyto==null)return false;
    try{
      mimeObj.setRecipients(javax.mail.Message.RecipientType.CC,
                                     (Address[])InternetAddress.parse(copyto));
      return true;
    }
    catch(Exception e)
    { return false; }
  }


public void setNeedAuth(boolean need) {
        //设置 SMTP 身份认证
        if(props==null)
```

```
            props=System.getProperties();

        if(need){
          props.put("mail.smtp.auth","成功");
        }else{
          props.put("mail.smtp.auth","失败");
        }
}
public boolean sendMail(){
      //发送邮件
   if(props==null)
     props=System.getProperties();                          //获得系统属性对象
    props.put("mail.smtp.host",getSmtpHost());              //设置 SMTP 主机
    Session session;
    session=Session.getDefaultInstance(props,null);         //获得邮件会话对象
    mimeObj=new MimeMessage(session);               //创建 MIME 邮件对象
    mp=new MimeMultipart();                           //创建 MimeMultipart 对象,保存邮件所有信息
    setNeedAuth(true);
   if(! from.equals("")) addFrom();
   else
        return false;
   if(! to.equals("")) addTo();
   else
        return false;
   if(! copyto.equals(""))
        addCopyto();
   if(! subject.equals(""))
        addSubject();
   if(! message.equals(""))
        addMessage();
   if(! attachedFile.equals(""))
        addAttachedFile();

   try{
     mimeObj.setContent(mp);
     //设置邮件内容
     mimeObj.saveChanges();
     //保存邮件
     Session mailSession=Session.getInstance(props,null);
     //建立邮件会话
     Transport transport=mailSession.getTransport("smtp");
     //创建发送对象
     transport.connect((String)props.get(getSmtpHost()),getUsername(),getPassword());
     //连接 SMTP 服务器
     transport.sendMessage(mimeObj,
```

```
                    mimeObj.getRecipients(javax.mail.Message.RecipientType.TO));
        //发送邮件
        transport.close();
        //关闭发送对象
        return true;
      }
      catch(Exception e){
        System.err.println("邮件发送失败!"+e);
        return false;
      }
  }
    /*
    * 收邮件处理
    */
  public String getPop3Host(){
        //获取 Pop3 主机名
        return pop3Host;
  }
  public void setPop3Host(String pop3Host){
        //设置 pop3 主机
        this.pop3Host=pop3Host;
  }
  public String receiveMail(String pop3Host){
        //收邮件
         String mailcontent="";
         Store store=null;
         Folder folder=null;
         try{
              Properties props=System.getProperties();
              Session session=Session.getDefaultInstance(props, null);
              //建立收邮件会话
              store=session.getStore("pop3");
              //准备收邮件
              store.connect(pop3Host, username, password);
              //连接 pop3 服务器
              folder=store.getDefaultFolder();
              //获取收件箱
              if(folder==null)
                   throw new Exception("没有邮箱");
              folder=folder.getFolder("inbox");
              if(folder==null)
                   throw new Exception("no pop3 inbox");
              folder.open(folder.READ_ONLY);
              //收件箱设置成只读
              Message[] msgs=folder.getMessages();
```

```
            //读取信件
            int getMaillen=0;
            if(msgs.length<5)
              getMaillen=5;
            else
              getMaillen=msgs.length;
             for(int msgnum=0; msgnum <getMaillen; msgnum++) {
                 mailcontent=mailcontent+getMailMessage(msgs[msgnum])+"\n\n\n\n";

             }
        }
        catch (Exception ex) {
             ex.printStackTrace();
        }
        finally {
             try {
                 if (folder!=null) folder.close(false);
                 if (store!=null) store.close();
             }
             catch (Exception ex2) {
                 ex2.printStackTrace();
             }
        }
        return mailcontent;
    }
public String getMailMessage(Message message) {
        //收取邮件的具体内容
        String mails=null;
        try {
             String fromadr=((InternetAddress)message.getFrom()[0]).getPersonal();
             if(fromadr==null)
               fromadr=((InternetAddress)message.getFrom()[0]).getAddress();
             //获取发邮件地址;
             mails="from: "+fromadr;
             String subject=message.getSubject();
             mails=mails+"\n"+"subject: "+subject;
             //获取邮件主题
             Part multiPart=message;
             Object content=multiPart.getContent();
             if(content instanceof Multipart) {
                 multiPart=((Multipart)content).getBodyPart(0);
                 mails=mails+"\n"+"[ multipart message ]";
             }
             //获取邮件内容
             mails=mails+"\n"+"content: "+content.toString();
```

```
            String contenttype=multiPart.getContentType();
            mails=mails+"\n"+"content:"+contenttype;
            if(contenttype.startsWith("text/plain") || contenttype.startsWith("text/html")) {
                //判断邮件类型
              InputStream is=multiPart.getInputStream();
                BufferedReader reader=new BufferedReader(new InputStreamReader(is));
                String thisline=reader.readLine();
                while(thisline!=null) {
                    mails=mails+"\n"+thisline;
                    thisline=reader.readLine();
                }
            }
        }
        catch(Exception ex) {
            ex.printStackTrace();
        }
        return mails;
    }
}
```

练习 1：用 JavaBean 发邮件。

请仔细阅读 MailBean. java，利用该 JavaBean 实现 SMTP 发邮件。运行结果如图 12-10～图 12-12 所示。



图 12-10　登录界面



图 12-11　设置收发邮件服务器



图 12-12　发邮件界面

练习 2：利用 JavaBean 接收邮件。

本次练习是利用 MailBean.java 实现 POP3 邮件的收取。具体要求如下。

(1) 仔细阅读 MailBean.java 中的 receiveMail()方法。了解 JavaMail API 实现 POP3 收邮件的主要应用。请修改程序清单 12-12 的 handreceiveMail.jsp 程序，观察运行结果。

程序清单 12-12：

```
<%@page contentType="text/html; charset=utf-8" language="java" import="beans.mail.*" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<!--handlereceiveMail.jsp-->
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>收邮件</title>
</head>
<body>
<jsp:useBean id="mailbean" scope="session" class="beans.mail.MailBean" />
<%
  mailbean.setPop3Host("pop3.sina.com.cn");
  //设置 POP3 主机,请根据实际情况修改
  mailbean.setUsername("xxx@sina.com.cn");
  //设置登录邮件名,请根据实际情况修改
  mailbean.setPassword("123456");
  //设置登录密码,请根据实际情况修改
  out.println("收到的邮件:<br/>");
  out.println(mailbean.receiveMail(mailbean.getPop3Host()));
```

```
%>
</body>
</html>
```

(2) 运行上述的 handlereceiveMail.jsp,可以输出 POP3 账号收件箱 5 封邮件内容,请修改该 JavaBean,实现 POP3 收邮件,并将收取的邮件放置在数据库中保存,要求每次只能收取新邮件(提示,可以考虑用邮件的标记来确定是否是新邮件)。

第 13 章　JSP 的 Servlet 编程

Servlet 是早于 JSP 出现的一种服务器端的开发技术。由于具有良好的业务逻辑处理能力,常作为控制器,与 JavaBean 和 JSP 结合形成 JSP 第二种 MVC 开发模式。Servlet 具有执行效率高,继承了 Java 语言的平台无关性、功能强大等特点。因此在很多 Web 应用领域中得到应用。本章简略介绍 Servlet 编程的要点以及设计与 Servlet 编程相关的实验。

13.1　预 备 知 识

Servlet 是应用标准 Servlet API 开发的 Java 程序。Servlet 可扩展服务器的功能,实现强大的 Web 应用。在本节中主要了解开发 Servlet 程序的常见 Servlet API 应用程序接口、Servlet 的开发与部署以及 JSP 的两种 MVC 开发模式。

13.1.1　常见的 Servlet API

Servlet API 是一个标准的 Java 扩展程序接口,主要由两个包: javax. servlet 和 javax. servlet. http 所构成。javax. servlet 包定义了针对开发客户自定义协议应用的类和接口,具体内容见表 13-1。javax. servlet. http 包主要定义了应用于 HTTP 协议相关的类和接口,具体内容见表 13-2。这两个扩展包也构成了 Servlet 的基本框架。Servlet 程序实际上就是 Java 程序,只是它必须要应用 Servlet API 来实现功能。

无论是实现用户自定义协议的 Servlet 程序还是实现 HTTP 协议的 Servlet 程序,都必须具有一个生命周期。Servlet 的生命周期分成 3 个阶段: 初始化阶段、响应用户请求阶段和终止阶段。

(1) 初始化阶段。实现了 Servlet 装载到 Servlet 容器中,Servlet 容器会创建一个 Servlet 对象,并调用 init()方法对该对象进行初始化。

表 13-1　javax. servlet 包

类 和 接 口	描　　述
Servlet	开发 Servlet 的接口
RequestDispatcher	抽象接口,创建接受用户的请求,并发送请求到其他资源的
ServletRequest	定义请求对象,封装请求信息
ServletResponse	定义响应对象,封装响应信息
ServletConfig	Servlet 容器为实现 Servlet 初始化传递信息
ServletContext	定义 Servlet 程序与 Servlet 容器通信的上下文
GenericServlet	抽象类,为开发独立协议应用的 Servlet 类提供支持
ServletInputStream	类,输入流用于读取客户传递的信息
ServletOutputStream	类,输出流向客户发送信息

表 13-2　javax. servlet. http 包

类和接口	描　　述
HttpServlet	抽象类,定义与 HTTP 协议相关的 Servlet,是 GenericServlet 的子类
HttpServletRequest	是 ServletRequest 的扩展,封装 HTTP 协议的请求
HttpServletResponse	是 ServletResponse 的扩展,封装 HTTP 协议的响应
HttpSession	定义实现 Session 机制的一个接口
Cookie	定义 Cookie 的类

(2) 响应用户请求阶段。Servlet 容器根据用户请求,生成 ServeltRequest 对象和 ServletResponse 对象,分别封装请求信息和响应信息。然后调用 Servlet 的 service()方法,接收 ServeltRequest 和 ServletResponse 对象,并执行 Servlet 的处理。

(3) Servlet 终止阶段。Servlet 容器调用 destroy()方法清除 Servlet 对象,释放 Servlet 占据的空间。

13. 1. 2　Servlet 的开发与部署

根据应用范围不同,可开发的 Servlet 类分成两种。

(1) 实现用户自定义的 Servlet 类,它必须是 javax. servlet. GenericServlet 类的子类,在编写这种 Servlet 类是,必须用 extends 扩展 GenericServlet。javax. servlet. GenericServlet 的常见方法见表 13-3。

表 13-3　javax. servlet. GenericServlet 类的常见方法

方　　法	描　　述
void init()	初始化 Servlet 对象
void service(ServletRequest, ServletResponse)	执行 Servlet 的请求,并响应
void destroy()	清除 Servlet
ServletConfig getServletConfig()	获取 ServletConfig 对象
ServletContext getServletContext()	获取 ServletContext 对象
java. lang. String getServletInfo()	获取 Servlet 的相关信息

(2) 实现 HTTP 协议的 Servlet 类。这种 Servlet 类是扩展 HttpServlet 的子类。javax. servlet. http. HttpServlet 的常见方法见表 13-4。

表 13-4　javax. servlet. http. HttpServlet 类的常见方法

方　　法	描　　述
protected void doGet(HttpServletRequest, HttpServletResponse)	服务器调用处理 Get 请求
protected void doPost(HttpServletRequest, HttpServletResponse)	服务器调用处理 Post 的请求
protected void doDelete(HttpServletRequest, HttpServletResponse)	服务器调用处理删除的请求
protected void doPut(HttpServletRequest, HttpServletResponse)	服务器调用处理 Put 的请求

续表

方　　法	描　　述
protected void doTrace(HttpServletRequest, HttpServletResponse)	服务器调用处理 Trace 的请求
protected void doHead(HttpServletRequest, HttpServletResponse)	从 service()方法中获取 HTTP 头,并处理这个请求
protected void doOption(HttpServletRequest, HttpServletResponse)	服务器调用处理 Option 的请求
protected void service(HttpServletRequest, HttpServletResponse)	接受 HTTP 请求,转发请求到 doXXX()系列方法处理请求

编写好一个 Servlet 程序后,需要成功编译生成 class 文件后,就要进入部署阶段。Servlet 的部署实际上就是将 Servlet 部署成为一个 Web 应用。需要为对应的 Web 应用的 web. xml 中定义 Servlet 以及 Servlet 的映射。其中 web. xml 中关于 Servlet 映射的常见标记见表 13-5。

表 13-5　web. xml 部署 Servlet 的常见标记

标　记	描　　述
servlet	定义 Servlet 元素
servlet-name	定义 Servlet 的名字
servlet-class	指定 Servlet 的类
jsp-file	将 jsp 文件配置成 Servlet,注意不能与 servlet-class 元素同时使用
load-on-startup	设置 Servlet 启动的优先级
init-param	设置 Servlet 的初始化参数
param-name	指定 Servlet 的参数名,必须与 init-param 联合使用
servlet-mapping	设置 Servlet 的映射
url-pattern	指定 Servlet 映射的 url 模式,可以出现多次,必须与 servlet-mapping 元素中出现

13. 1. 3　JSP 的两种开发模式的比较

Sun Microsystems 公司发布了 JSP 的两种开发模式:JSP＋JavaBean 和 JSP＋JavaBean＋Servlet。这两种工作模式实现了 MVC(模型-视图-控制)模型。

1. JSP＋JavaBean

JSP＋JavaBean 模式中,JSP 承担了 MVC 中的控制器和视图的部分,JSP 接受用户的请求,以及处理的响应结果,实现流程的控制以及内容的显示。对于 JavaBean 承担着模型的功能,实现对数据的访问和处理。JSP 和 JavaBean 之间的关系见图 13-1。

2. JSP＋JavaBean＋Servlet

JSP＋JavaBean＋Servlet 模式中,JSP 作为视图,控制器部分从 JSP 中脱离出来,由 Servlet 来承担。至于 JavaBean 仍充当模型的部分,实现数据部分的处理。JSP、JavaBean 和 Servlet 之间的关系见图 13-2。

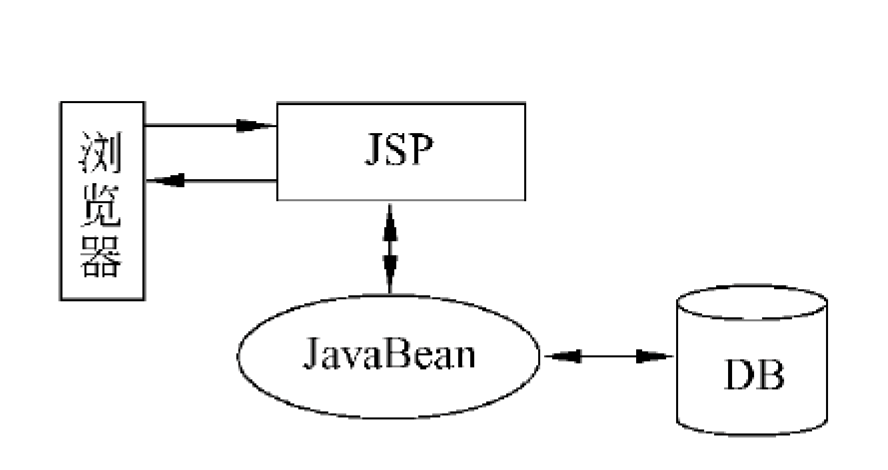


图 13-1　JSP＋JavaBean 工作模式

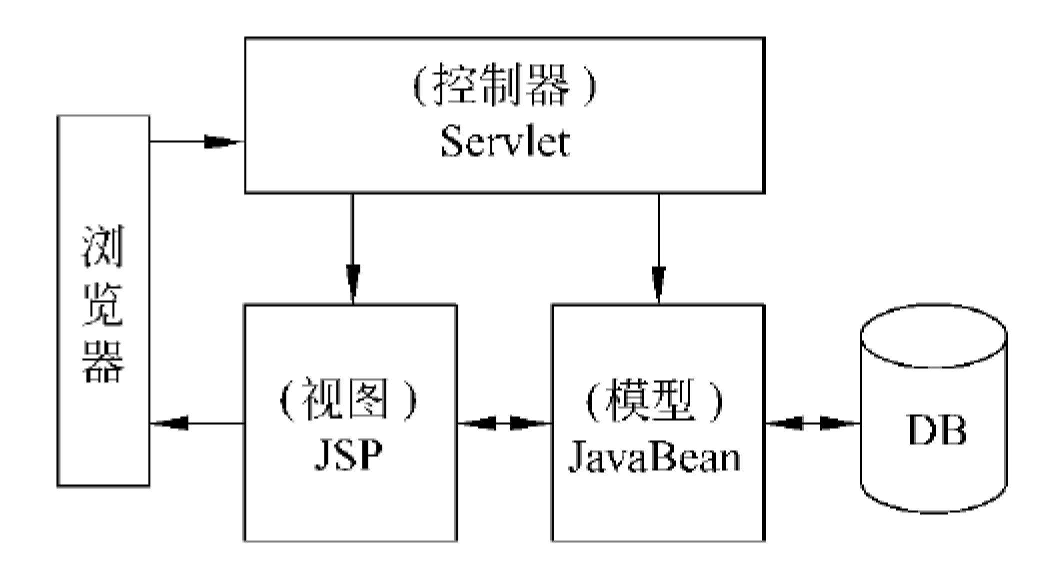


图 13-2　JSP＋JavaBean＋Servlet 工作模式

3. 二者的比较

JSP＋JavaBean 中,JSP 承担了控制器和视图的两种功能。这导致了 JSP 页面中存在大量的 Java 代码,页面难以维护。但是在开发小型 Web 应用上存在灵活开发简单的优势。至于 JSP＋JavaBean＋Servlet 模型,MVC 各层次的关系更加清晰。Servlet 分担了 JSP 原来的控制器的角色,一方面使得 JSP 可以充分表现视图;另一方面使得 Servlet 充分体现处理业务逻辑的优势。JSP＋JavaBean＋Servlet 模式广泛应用在复杂大型的 Web 应用中。

13.2　实验 13.1　Servlet 的开发和部署

实验目的:

(1) 了解开发 Servlet 的全过程。

(2) 了解开发用户自定义协议的 Servlet 和 HTTP 协议的 Servlet 的不同。

(3) 了解和熟练部署 Servlet 的 web.xml。

(4) 了解和掌握 HttpServlet 实现处理客户端的请求。

实验内容:

(1) 开发 Servlet,要求实现一个计数器,计算访问该 Servlet 的次数。要求:分别用用户自定义协议的 Servlet 和实现 HTTP 协议的 Servlet 来实现这样的一个信息的输出。并定义对应 Web 应用的 web.xml,实现对 Servlet 的部署,以及运行部署后的 Servlet。

(2) 处理表单数据的 Servlet。要求:设计一个可以显示个人简历信息的 Servlet。其中,个人简历的信息由用户从表单中设置。

实验步骤:

练习 1:用 Servlet 实现计数器。

本练习是要求开发不同类型的 Servlet,实现计数器,来计算访问 Servlet 的次数。具体要求如下:

(1) 下列程序清单 13-1 定义了 CounterServlet.java 程序,仔细阅读该程序,判断该 Servlet 是否可以计算访问次数。编译该 Java 程序,将编译后的程序保存在 WEB-INF\classes 目录下。程序清单 13-2 定义了对应 Web 应用的 web.xml 的部分代码,请将程序清

单 13-2 中 代码 1 ～ 代码 4 处补充完整，保存在 WEB-INF 目录下然后在 Tomcat 服务器中运行该 Servlet，并记录运行结果。

程序清单 13-1：

```
//CounterServlet.java
package Servlet;
import javax.servlet.*;
import java.io.*;

public class CounterServlet extends GenericServlet {
    private static int counter=0;
    final static String f="/tmp/counter.tmp";

    public void init () throws ServletException {
      try {
        FileInputStream    in=new FileInputStream(f);
        InputStreamReader reader=new InputStreamReader(in);
        counter=Integer.parseInt(new BufferedReader(reader).readLine());
        in.close();
      } catch (IOException e) {
         System.out.println(e.getMessage());
      }
    }

    public void destroy() {
      try {
        FileOutputStream out=new FileOutputStream(f);
        PrintStream        ps=new PrintStream(out);
        ps.println(counter);
        out.close();
      } catch (IOException e) {}
      super.destroy();
    }

    public void service(ServletRequest request, ServletResponse response)
                          throws ServletException, IOException {
      PrintWriter out=response.getWriter();
      increaseCounter();
      response.setContentType("text/html; charset=GB2312");
      out.print("<html><body>");
      out.println("<p align='center'>");
      out.println("共访问"+counter+"次");
      out.println("</p>");
      out.println("</body>");
      out.println("</html>");
```

```
        out.close();
    }

    synchronized void increaseCounter() {
        counter++;
    }
}
```

程序清单 13-2：

```
<?xml version="1.0" encoding="gb2312"?>
<!--web.xml-->
<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
    version="2.4">
  <description>计算访问次数</description>

  <servlet>
      <servlet-name>Counter</servlet-name>
      <servlet-class> 代码 1 </servlet-class>          <!--请设置 Servlet 类 -->

  </servlet>
       代码 2                                           <!--请配置 Servlet 映射 -->
      代码 3                                            <!--请设置 Servlet 名-->
  <url-pattern>/Counter</url-pattern>
     代码 4                                             <!--请结束 Servlet 映射-->
</web-app>
```

（2）将程序清单 13-1 改写成 HttpServlet 的程序 CounterHttpServlet.java，代码见程序清单 13-3，编译该程序，并设置对应的 web.xml。运行该 Servlet，观察并记录运行结果，比较该运行结果与 CounterServlet 的运行结果是否一致？如果运行结果不一致，请修改程序 CounterHttpServlet.java，并解释 CounterHttpServlet 运行结果与上述的 CounterServlet 的不同的原因。如果运行结果一致，请分别解释二者运行的工作原理。

程序清单 13-3：

```
//CounterHttpServlet.java
package Servlet;
import javax.servlet.http.*;
import java.io.*;

public class CounterHttpServlet {
    private static int counter=0;
    final static String f="/tmp/counter.tmp";
```

```
    public void doPost(HttpServletRequest req,HttpServletResponse res){
        try {
            FileInputStream  in=new FileInputStream(f);
            InputStreamReader reader=new InputStreamReader(in);
            counter=Integer.parseInt(new BufferedReader(reader).readLine());
            in.close();
            PrintWriter out=res.getWriter();
            counter++;
            res.setContentType("text/html; charset=GB2312");
            out.print("<html><body>");
            out.println("<p align='center'>");
            out.println("共访问"+counter+"次");
            String str="中文";
            out.println(new String(str.getBytes("ISO-8859-1"),"gb2312"));
            out.println("</p>");
            out.println("</body>");
            out.println("</html>");

            out.close();

            FileOutputStream output=new FileOutputStream(f);
            PrintStream      ps=new PrintStream(output);
            ps.println(counter);
            out.close();
         } catch (IOException e) {
             System.out.println(e.getMessage());
         }
    }

   public void doGet(HttpServletRequest req,HttpServletResponse res){
        doPost(req,res);
    }

}
```

练习 2：用 HttpServlet 实现客户端的表单数据。

已知有 resume.jsp 文件提供了设置个人简历的简单表单，见程序清单 13-4。请将程序清单 13-5 的 ResumeServlet.java 的 代码 1 ～ 代码 3 和 代码段 1 ～ 代码段 2 处补充完整，实现显示用户的个人简历信息。运行结果如图 13-3 和图 13-4 所示。

程序清单 13-4：

```
<!--resume.jsp-->
<%@page contentType="text/html; charset=gb2312" language="java" import="java.sql.*" %>
```

设置个人简历

个人信息

姓　　名：黄海
性　　别：◉男　○女
出生日期：1990年-10月-01日
通讯地址：北京市111信箱
电　　话：010-99999999
E - mail：huanghai @yahoo.com.cn

教育背景

毕业学校：南京大学
学　　历：硕士
专　　业：计算机应用
专业描述：
计算机软件的开发和网络应用

求职意向

期望工作性质：◉全职　○兼职
期望工作职位：软件工程师
期望工作地点：南京
期待工作报酬：面谈
确认

图 13-3　设置个人简历



图 13-4　显示设置后的个人简历

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>定制个人简历</title>
```

```
</head>

<body>
<fieldset>
<legend>设置个人简历</legend>
<form method="post" action="ResumeServlet">
  <h3>个人信息</h3><!--定义个人信息  -->
  姓     名:
  <input type="text" name="user_name"  maxlength="30"/>  
  <br/>
  性     别:
  <input type="radio" name="user_gender" value="男"/>男  
  <input type="radio" name="user_gender" value="女"/>女
  <br/>
  出生日期:
  <input type="text" name="user_birth" value="1990年-01月-01日" /><br/>
  通讯地址:
  <input type="text" name="user_addr" maxlength="30"/><br/>
  电     话:
  <input type="text" name="user_tele" maxlength="30"/><br/>
  E - mail:
  <input type="text" name="user_emname" maxlength="30"/>@
  <input type="text" name="user_emdns" maxlength="20"/>

  <h3>教育背景</h3>
  <!--定义教育背景-->
  毕业学校:
  <input type="text" name="edu_school" maxlength="40"/><br/>
  学     历:
    <select name="edu_degree">
         <option value="中专">中专</option>
         <option value="大专">大专</option>
         <option value="本科">本科</option>
         <option value="硕士">硕士</option>
         <option value="博士">博士</option>
        </select>
        <br/>
专     业:
<input type="text" name="edu_major"/><br/>
专业描述:<br/>
<textarea name="edu_des" rows="6" cols="30">
</textarea>
  <br/>

<h3>求职意向</h3>
```

```
期望工作性质：
<input type="radio" name="job_type" value="全职"/>全职
<input type="radio" name="job_type" value="兼职"/>兼职
<br/>
  期望工作职位：
<input type="text" name="job"/>
<br/>
  期望工作地点：
<input type="text" name="job_place"/>
<br/>
  期待工作报酬：
<input type="text" name="job_salary"/>
<br/>
  <input type="submit" value="确认"/>
</form>
</body>
</html>
```

程序清单 13-5：

```
//ResumeServlet.java
package Servlet;
import javax.servlet.http.*;
import javax.servlet.*;

import java.io.*;
import java.util.*;
public class ResumeServlet extends HttpServlet {
        String user_name;                    //用户姓名
        String user_gender;                  //用户性别
        String user_birth;                   //用户出生日期
        String user_addr;                    //用户的地址
        String user_tele;                    //用户的电话
        String user_email;                   //用户的 E-mail;

        String edu_school;                   //毕业学校
        String edu_degree;                   //学位
        String edu_major;                    //专业
        String edu_des;                      //专业描述

        String job_type;                     //工作性质
        String job;                          //工作
        String job_place;                    //工作地点
        String job_salary;                   //期望工资
```

```
public void doPost(代码 1,代码 2) throws 代码 3, IOException {        //请补充代码
    request.setCharacterEncoding("GB2312");
    response.setContentType("text/html;charset=GB2312");
    PrintWriter out=response.getWriter();
    setPersonalInfo(request);                    //设置个人信息
    setEduInfo(request);                         //设置教育背景信息
    setJobInfo(request);                         //设置期望工作信息
    if(user_name.equals(""))
        displayError(out);
    else
        displayResume(out);
}

public void displayResume(PrintWriter out){
//输出定制后的简历
out.println("<html>");
    out.println("<head>");
    out.println("<link rel='stylesheet' type='text/css' href='style.css' />");
    out.println("</head>");
out.println("<body>");
displayPersonInfo(out);
displayEduInfo(out);
displayJobInfo(out);
out.println("</body></html>");
}

public void displayError(PrintWriter out){
//显示简历信息错误
out.println("<html>");
    out.println("<head>");
    out.println("<link rel='stylesheet' type='text/css' href='style.css' />");
    out.println("</head>");
out.println("<body>");
out.println("请输入正确信息");
out.println("</body></html>");
}

public void setPersonalInfo(HttpServletRequest request){
//设置个人信息
    user_name=request.getParameter("user_name");
    user_gender=request.getParameter("user_gender");
    user_birth=request.getParameter("user_birth");
    user_addr=request.getParameter("user_addr");
```

```
        user_tele=request.getParameter("user_tele");
        String email1=request.getParameter("user_emname");
        String email2=request.getParameter("user_emdns");
        if(user_name==null) user_name="";
        if(user_gender==null)user_gender="";
        if(user_birth==null) user_birth="";
        if(user_addr==null)user_addr="无";
        if(user_tele==null)user_tele="无";
        if(email1==null || email2==null) user_email="无";
}

  public void displayPersonInfo(PrintWriter out){
  //输出个人信息
              代码段 1                                    //请补充代码段
  }

  public void setEduInfo(HttpServletRequest request){
  //定制教育背景
        edu_school=request.getParameter("edu_school");
        edu_degree=request.getParameter("edu_degree");
        edu_major=request.getParameter("edu_major");
        edu_des=request.getParameter("edu_des");
        if(edu_school==null)edu_school="无";
        if(edu_degree==null)edu_degree="";
        if(edu_major==null) edu_major="";
        if(edu_des==null)edu_des="无";
  }

  public void displayEduInfo(PrintWriter out){              //输出教育背景
     代码段 2                                               //请补充代码段
  }

  public void setJobInfo(HttpServletRequest request){
  //设置期望工作信息
        job_type=request.getParameter("job_type");
        job=request.getParameter("job");
        job_place=request.getParameter("job_place");
        job_salary=request.getParameter("job_salary");
        if(job_type==null)job_type="";
        if(job==null)job="";
        if(job_place==null) job_place="";
        if(job_salary==null)job_salary="面谈";
  }
```

```
        public void displayJobInfo(PrintWriter out){
        //显示期望工作信息
            代码段 3                    //请补充代码段
        }
    }
```

请将补充代码后的程序编译,并配置对应 Web 应用的 web.xml,然后启动 Tomcat 6.0,并运行 resume.jsp 和 ResumeServlet。观察运行结果。如果将程序清单 13-3 的 resume.jsp 发送表单的方式改写成 get,再运行上述两个程序,观察运行结果,会发生什么变化? 如果有变化,请解释原因。

13.3　实验 13.2　JSP 的两种开发模式

实验目的:

(1) 深入了解 JSP 的两种开发模式 JavaBean+JSP 和 JavaBean+Servlet+JSP。

(2) 比较 JavaBean+JSP 和 JavaBean+Servlet+JSP 的异同。

(3) 利用这两种开发模式的特点,熟练开发具有实际意义的 Web 应用。

实验内容:

本次实验的内容类似于第 12 章的第 12.3 节。也是模仿 ForumLite 2.1 的结构,用 JSP 的第二种开发模式 JavaBean+Servlet+JSP 开发一个具有最基本功能的简易论坛。功能要求:此系统要实现允许用户浏览论坛、发表新论点,也可以浏览并回复论坛的已有论点。论坛中所有的数据信息用 MySQL 数据库保存。具体功能要求如下:

(1) 任何用户可以直接发表论题,论题包括标题、内容、发表者的电子邮件及发表日期;如果有用户对该论题回复,要求能显示已回复论题的次数。

(2) 任何用户可以浏览已发表的论题,并可以针对特定的论题进行回复。回复信息包括标题、内容、回复者的电子邮件以及回复的时间。

(3) 要求所有论题的以分页的形式显示,每页面 10 个论题。

(4) 要求某论题的所有回复也以分页显示,每页面 10 个回复。

实验步骤:

按照实验内容提出的基本要求,请自行进行系统设计、数据库设计,并采用 JavaBean+Servlet+JSP 工作模式二的方式,编写相应的程序实现简易论坛。使论坛的运行结果类似图 13-5～图 13-8 所示。

问题: 请说出 JSP+JavaBean 模式与 JSP+JavaBean+Servlet 模式下承担业务逻辑处理各是由哪种技术来承担。比较你用这两种模式开发实现简易论坛的特点,并说明哪种模式更加适应简易论坛的开发,请说明你的理由。

FORUM 论坛

论题	回复	提交者	创建日期
New World	3	Chen Yi chenyi@yahoo.com	2007-04-12
JSP world	0	Hilda Follis hilda_follis@tyus.com	2007-05-09
JavaBean	1	Rita Cliff rita_cliff@yahoo.com	2007-05-09
Life is Beautiful	0	Tommy Lee tommy24543@yahoo.com	2007-05-16
Love is forever	0	Ammy Huannn ammy@sina.com	2007-05-16
Cartoon world	5	Jain Some jain@rui.com.ld	2007-05-16
Flying	0	Hu Haisong haisong@yehe.com	2007-05-16

页:
-1-

创建论题

留言人

图 13-5　分页显示论题

		rita_cliff@yahoo.com	
Life is Beautiful	0	Tommy Lee tommy24543@yahoo.com	2007-05-16
Love is forever	0	Ammy Huannn ammy@sina.com	2007-05-16
Cartoon world	5	Jain Some jain@rui.com.ld	2007-05-16
Flying	0	Hu Haisong haisong@yehe.com	2007-05-16

页:
-1-

创建论题

留言人

E-Mail

标题

内容

发送

模仿ForumLite编写的最小最简单的论坛。
© 2008年8月10日《Web开发技术实用教程》

图 13-6　发表论题界面

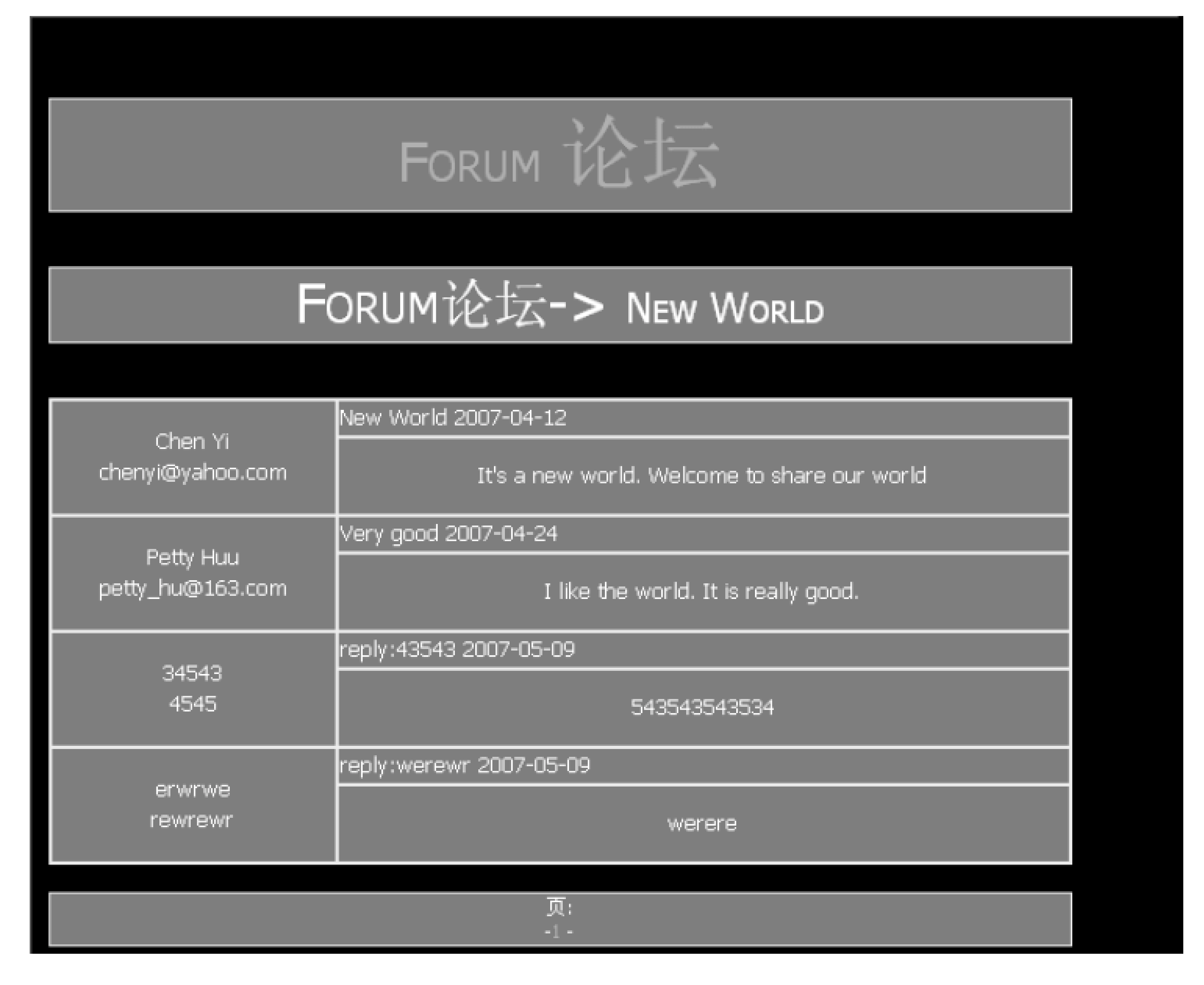


图 13-7　分页显示特定论题的界面

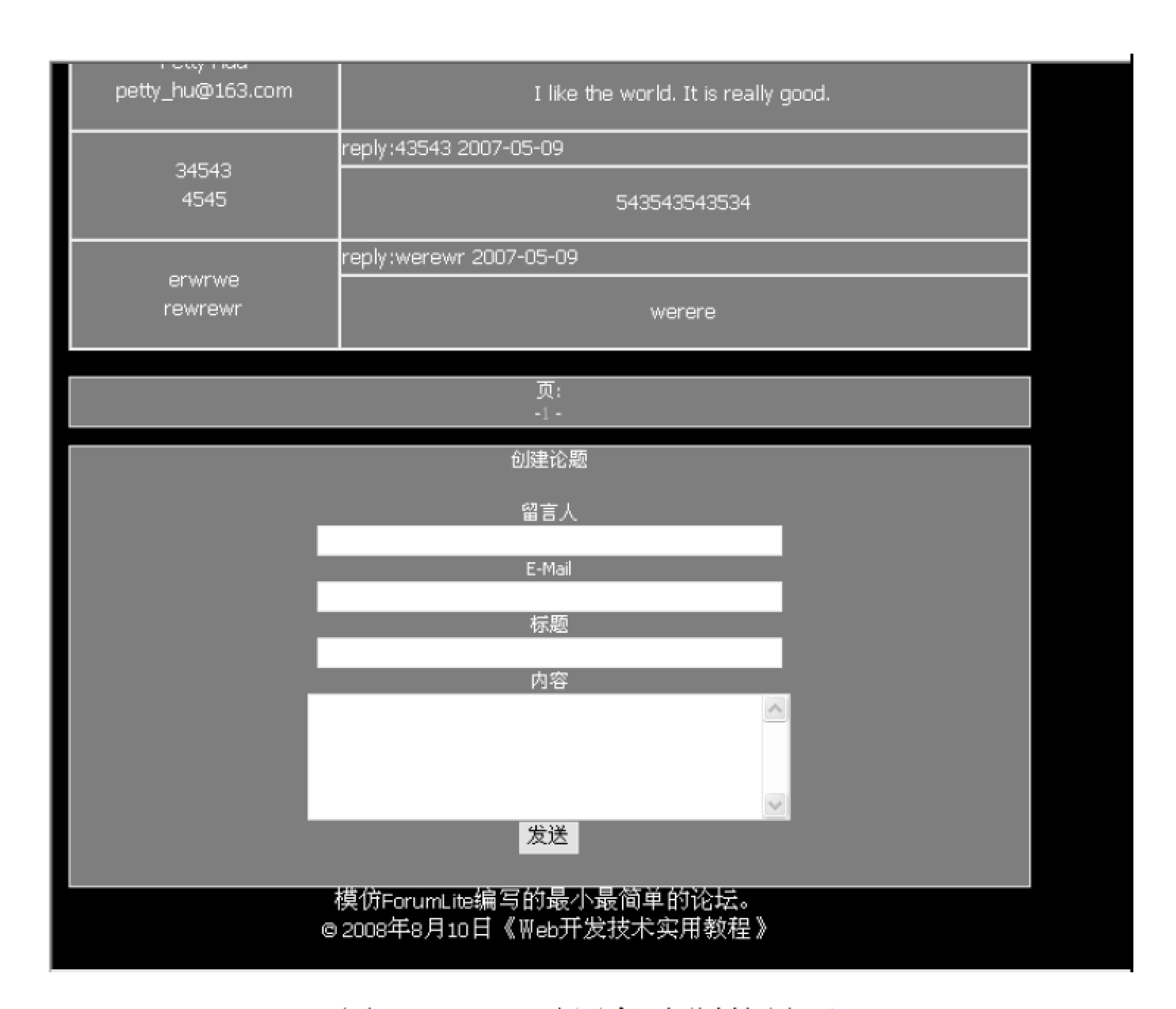


图 13-8　显示回复论题的界面

13.4　实验 13.3　Servlet 动态生成图像

实验目的：

(1) 了解 Servlet 技术在处理图形的应用。

(2) 初步了解用 Servlet 动态生成图像验证码的原理。

(3) 深入了解和运用 Servlet 技术解决实际问题。

实验内容：

（1）用 Servlet 生成图片。要求：用 Servlet 绘制一个简单图形。

（2）制作一个 Servlet 来绘制某小学某学期六个年级男女学生数学平均成绩的比较柱状图。

（3）用 Servlet 动态生成图像验证码。

实验步骤：

练习 1：用 Servlet 生成图片。

本次练习的主要目的是了解 Servlet 绘制图形的主要过程。具体要求：下列程序清单 13-6 中是 ImageServlet. java 的代码。请阅读该程序，了解该 Servlet 程序的主要功能，并分析它的运行结果是什么？然后编译该 Servlet 程序，部署并在 Tomcat 6. 0 中运行该 Servlet，记录运行结果，观察运行结果是否与你分析的结果是否一致，并记录 Servlet 绘制图形的主要流程。

程序清单 13-6：

```
//ImageServlet.java
package myimage;
import javax.servlet.* ;
import javax.servlet.http.* ;
import java.io.* ;
import java.awt.* ;
import java.awt.image.* ;
import com.sun.image.codec.jpeg.* ;
public class ImageServlet extends HttpServlet{

public void doGet(HttpServletRequest req, HttpServletResponse res)
  throws ServletException, IOException {
  doPost(req,res);
}

public void doPost(HttpServletRequest req, HttpServletResponse res)
  throws ServletException, IOException
{
  BufferedImage image=new BufferedImage(100,100, BufferedImage.TYPE_INT_RGB);
  //绘制图形
  Graphics g=image.getGraphics();
  g.setColor(Color.blue);
  g.fillRect(25,25,50,50);
  //绘制填充矩形
  //创建图形并输出
  res.setContentType("image/jpeg");
  JPEGImageEncoder encoder=JPEGCodec.createJPEGEncoder(res.getOutputStream());
  encoder.encode(image);
```

```
    }
}
```

练习 2：Servlet 绘制柱状图。

已知有一个 MySQL 数据库 StudentDB，在 StudentDB 中有数据表 math，具体表的结构和数据见图 13-9，其中 grade_id 表示编号，grade 表示年级，gender 表示性别，score 表示数学成绩。有 MathServlet.java 代码见程序清单 13-7，它可以访问数据表 math，比较并制作不同年级的男女学生的数学成绩的统计柱状图。

```
mysql> describe math;
+----------+------------------+------+-----+---------+----------------+
| Field    | Type             | Null | Key | Default | Extra          |
+----------+------------------+------+-----+---------+----------------+
| grade_id | int(10) unsigned | NO   | PRI | NULL    | auto_increment |
| grade    | varchar(8)       | NO   |     | NULL    |                |
| gender   | varchar(8)       | NO   |     | NULL    |                |
| score    | int(11)          | NO   |     | NULL    |                |
+----------+------------------+------+-----+---------+----------------+
4 rows in set (0.01 sec)

mysql> select * from math;
+----------+--------+--------+-------+
| grade_id | grade  | gender | score |
+----------+--------+--------+-------+
|        1 | 一年级 | 男     |    90 |
|        2 | 一年级 | 女     |    92 |
|        3 | 二年级 | 男     |    91 |
|        4 | 二年级 | 女     |    93 |
|        5 | 三年级 | 男     |    90 |
|        6 | 三年级 | 女     |    94 |
|        7 | 四年级 | 男     |    92 |
|        8 | 四年级 | 女     |    90 |
|        9 | 五年级 | 男     |    91 |
|       10 | 五年级 | 女     |    90 |
|       11 | 六年级 | 男     |    90 |
|       12 | 六年级 | 女     |    94 |
+----------+--------+--------+-------+
12 rows in set (0.01 sec)
```

图 13-9　MySQL 下数据表 math 的所有记录

仔细阅读 MathServlet.java，了解程序代码的作用，将程序清单 13-7 中 代码 1 ～ 代码 4 处的空白，补充完整并解释 4 处的作用，实现对数据表的访问和统计，并将统计结果通过统计柱状图表示出来，使运行结果如图 13-10 所示。

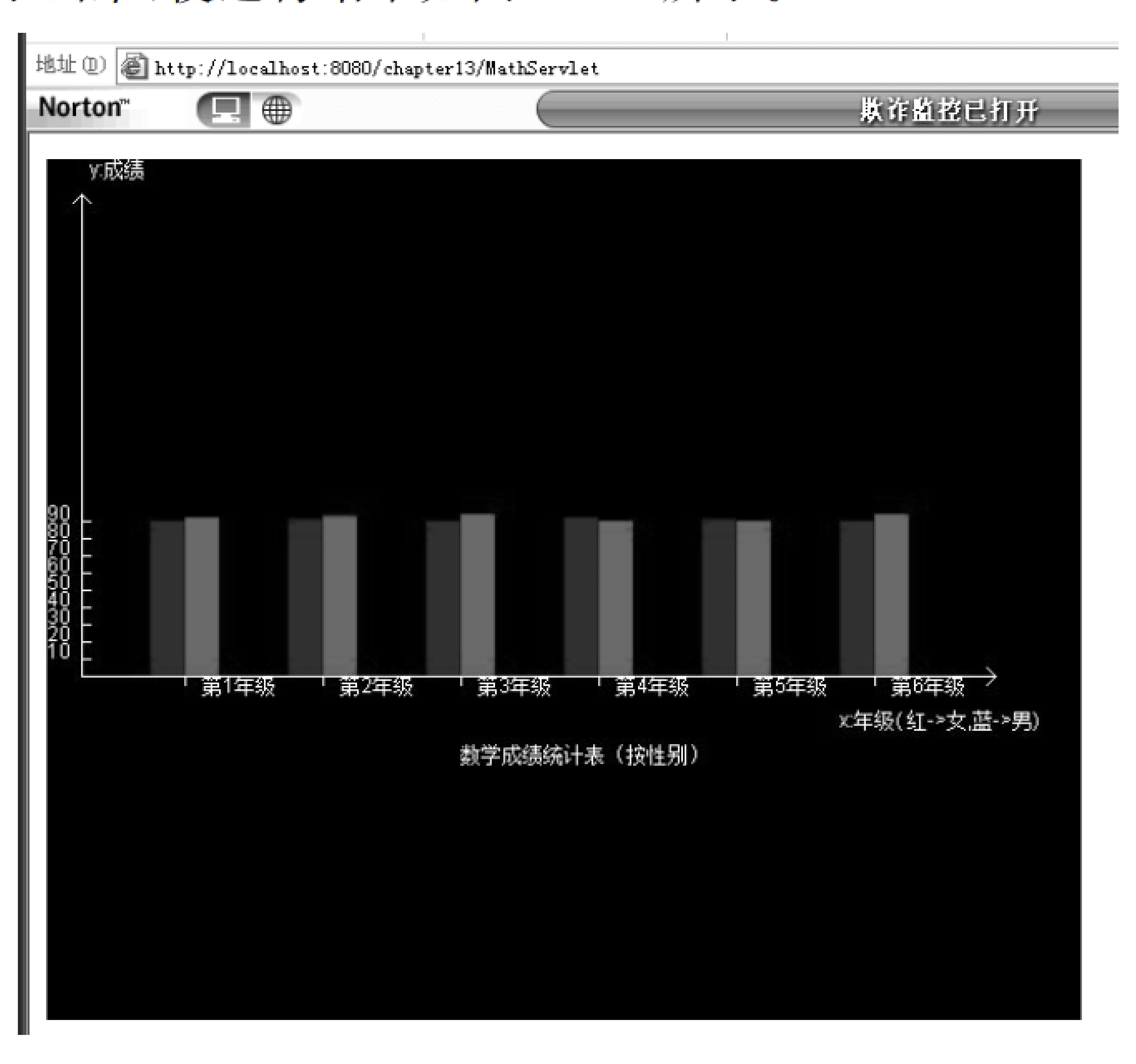


图 13-10　统计柱状图

程序清单 13-7：

```
//MathServlet.java
package Servlet;
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.awt.*;
import java.awt.image.*;
import com.sun.image.codec.jpeg.*;
import java.sql.*;
public class MathServlet extends HttpServlet{

/**
* 该程序绘制 6 个年级按性别区分的数学成绩统计表
**/
static int[][]grade=new int[6][2];
public void doGet(HttpServletRequest req, HttpServletResponse res)
  throws ServletException, IOException {
  doPost(req,res);
}

  public void doPost(HttpServletRequest req, HttpServletResponse res)
  throws ServletException, IOException {
  res.setCharacterEncoding("GB2312");
  try{
      getDataFromTable();
  }catch(SQLException e){
      e.printStackTrace();
  }

  BufferedImage image=new BufferedImage(600,500, BufferedImage.TYPE_INT_RGB);
  //创建绘图区
  Graphics g=image.getGraphics();
  g.setColor(Color.black);
  g.fillRect(0,0, 600,500);
  g.setColor(Color.WHITE);
  g.drawLine(20,20,20,300);
  for(int i=1;i<10;i++){
  //绘制 y 轴
     g.drawLine(20,300-i*10,20+5,300-i*10);
     g.drawString(""+(i*10), 0, 300-i*10);
  }
  for(int i=1;i<7;i++){
  //绘制 x 轴
     g.drawLine(80*i,300,80*i,305);
```

```
            g.drawString("第"+i+"年级",10+80*i,310);
        }
        g.drawLine(15,25,20,20);
        g.drawLine(20, 20, 25, 25);
        //绘制箭头
        g.drawLine(20, 300, 550, 300);

        g.drawLine(545,295,550,300);
        g.drawLine(550, 300, 545, 305);
        g.setColor(Color.yellow);
        g.drawString("y:成绩", 25, 10);
        g.drawString("x:年级 (红->女,蓝->男)",460,330);
        for(int i=0;i<6;i++){
            for(int j=0;j<=1;j++){
                if(j==0) g.setColor(Color.blue);
                else g.setColor(Color.red);
                g.fillRect(80*(i+1)+20*(j==0?-1:0), 300-grade[i][j], 20, grade[i][j]);
            }
        }
        g.setColor(Color.white);
        g.drawString("数学成绩统计表 (按性别)",240,350);
        res.代码 1;                                                    //补充代码
        JPEGImageEncoder encoder=代码 2;                                //补充代码
        encoder.encode(image);
    }
    public void getDataFromTable()throws SQLException{
        try{
            Class.forName("代码 3").newInstance();                       //补充代码
            Connection con=java.sql.DriverManager.getConnection(代码 4);  //补充代码
            Statement stmt=con.createStatement();
            ResultSet rs=stmt.executeQuery("select * from math");
            int gr,ge;
            //gr 表示年级,ge 表示性别
            ge=0;
            gr=0;
            while(rs.next()){
            //将各年级男女学生的平均分保存在数组中
                grade[ge][gr]=rs.getInt("score");
                gr=(gr+1)%2;
                if(gr==0)
                    ge++;
                if(ge>5) throw new SQLException("警告!出现严重错误!");
            }
        }catch(Exception e1){
```

```
            e1.printStackTrace();
        }
        }

}
```

练习 3：Servlet 动态生成图像验证码。

在网站填写表单，例如填写登录表单，往往需要填写验证码。验证码是随机生成的字符串，以某种方式保存起来，例如保存在 session 中。用户每次访问网站，都需要比较验证码是否一致。验证码的出现，有效地防止其他网站盗链资源。Servlet 技术的动态生成图像的能力，为生成图像验证码的验证功能的实现提供了保证。本次练习的要求如下。

(1) 已知程序清单 13-8 定义 CheckCodeServlet.java，请将程序的方法 createRandom()的 代码段 补充完整，使得 createRandom()方法实现动态生成 4 个随机字符(字符的范围是数字 0～9，大小写 26 个英文字母)。

程序清单 13-8：

```
//CheckCodeServlet.java
package Servlet;
import javax.servlet.* ;
import javax.servlet.http.* ;
import java.io.* ;
import java.util.* ;
import com.sun.image.codec.jpeg.* ;
import java.awt.* ;
import java.awt.image.* ;
public class CheckCodeServlet extends HttpServlet {
//处理 get 请求
    Random r;
    //定义随机对象
  public void doGet(HttpServletRequest request, HttpServletResponse response) throws
      ServletException, IOException {
    response.setCharacterEncoding("GB2312");
    response.setContentType("image/jpeg");
    //必须设置 ContentType 为 image/jpeg

    response.setHeader("Pragma", "no-cache");
    //禁止图像缓存
    response.setHeader("Cache-Control", "no-cache");
    response.setDateHeader("Expires", 0);

    Date d=new Date();
    long lseed=d.getTime();
    //利用时间生成随机种子
    r=new Random(lseed);
```

```
    //设置随机种子

    BufferedImage bi=new BufferedImage(60,32, BufferedImage.TYPE_INT_RGB);
    Graphics2D g=bi.createGraphics();
    g.clearRect(0, 0, 60, 32);
    g.setColor(Color.WHITE.brighter());
    g.setFont(new Font("Times New Roman",Font.PLAIN,18));
    String codes=createRandom();                          //生成随机字符
    g.drawString(codes, 8, 20);
    HttpSession session=request.getSession();
    //将验证码保存在 session 中
    session.setAttribute("codes",codes);
    try {
    //使用 JPEG 编码,输出到 response 的输出流
      JPEGImageEncoder encoder=JPEGCodec.createJPEGEncoder(response.
                                                        getOutputStream());
      JPEGEncodeParam param=encoder.getDefaultJPEGEncodeParam(bi);
      param.setQuality(1.0f, false);
      encoder.setJPEGEncodeParam(param);
      encoder.encode(bi);
    }
    catch (Exception ex) {
    }
  }

  public String createRandom(){
    //获取随机编码
      代码段            //请补充代码,使得返回的 4 位随机编码由数字和英文字母构成
  }
}
```

(2) 编译并部署该 Servlet,编写对应的 web. xml 文件,设置 CheckCodeServlet。请写出 web. xml 的代码。

(3) 将部署后的 Servlet 插入到登录表单中,代码如程序清单 13-9 的 login. jsp,请将 login. jsp 的 代码 1 处补充完整,使之运行结果如图 13-11。

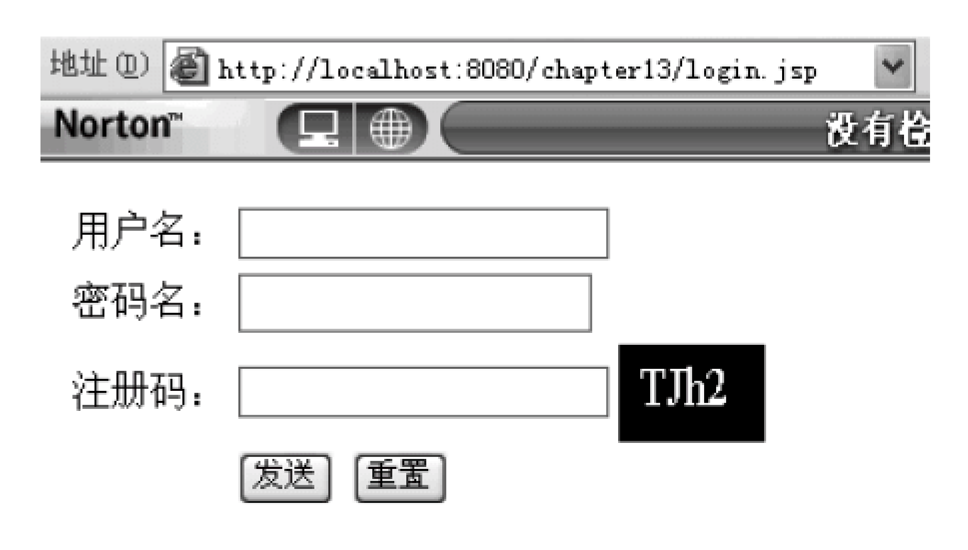


图 13-11　Servlet 动态生成图像验证码的运行结果

程序清单 13-9：

```
<!--login.jsp -->
<%@page contentType="text/html; charset=gb2312" language="java" import="java.sql.*" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
<title>登录</title>
</head>

<body>
<table>
<form method="post" action="CheckServlet">
<tr><td>用户名：</td>
    <td><input type="text" name="username" /></td>
    <td> </td>
</tr>
<tr>
    <td>密码名：</td>
    <td><input type="password" name="password" /></td>
    <td> </td>
<tr>
    <td>注册码：</td>
    <td><input type="text" name="code" /></td>
    <td><img 代码 1  width="60" height="40"/></td>      <!--添加注册码图片 -->
<tr>
<td> </td>
<td>
<input type="submit" value="发送"/>
<input type="reset" value="重置"/>
</td>
</form>
</table>
</body>
</html>
```

（4）请说明 Servlet 编程的特点，比较它与 JavaBean 的异同。

第 14 章　JSP 和 XML

在第 5 章介绍 XML 的基本语法内容和 XML 在客户端的相关应用。实际上，XML 广泛应用在服务器端，是服务器端的技术。JSP 与 XML 都是 Sun 公司的 J2EE 的重要组成，JSP 开发服务器端程序具有强大的优势，而 XML 在数据表达和系统之间传递数据具有极大的便利，二者的结合可以开发出功能强大的 Web 应用。在本章中，将从 JSP 生成 XML 文件、JSP 解析 XML 文件以及 JSP 的自定义标签等方面来进一步了解服务器端的开发。

14.1　预备知识

XML 是一种具有严格语法规则的良构性语言，具有良好的描述数据的特点。它是一种服务器端的常见技术。JSP 有 3 种方式可以使用 XML 数据。

（1）直接使用 XML 文件。

（2）利用 JavaBean 处理 XML 数据。

（3）利用 XML 实现 JSP 的自定义标签。

14.1.1　JSP 直接使用 XML 文件

JSP 使用 XML 数据，又有以下几种情况。

1. JSP 要直接嵌入 XML 数据，直接生成 XML 文件

需要通过＜％@page％＞指令说明文件的内容的类型，即指定 page 指令的 contentType 的属性为“text/xml”。形式如下：

```
<%@page contentType="text/xml"%>
```

然后，在该指令后面，直接嵌入 XML 数据。通过这种方式，JSP 可以将嵌入的 XML 数据生成 XML 文件。方便客户端或其他的应用获取对应的 XML 文件。在 JSP 中直接利用 XSLT 实现对 XML 数据的转换，达到有效的输出。或在 JSP 中通过非 XSLT 方式，实现对 XML 数据有效的表示或转换。

2. JSP 使用 XML 文件

JSP 也可以直接使用 XML 文件，利用 XML 数据为 JSP 指定具体的处理动作。在执行这样的功能时，往往需要对 XML 数据进行解析，获取有效的输出。JSP 可以利用 JAXP API 实现对 XML 数据的解析。对于 JAXP API 的相关介绍见配套教材的第 14.2.1 节。常见用于 Java 解析 XML 的还有 JDOM 和 DOM4J，这些 Java API 也是常见用于解析和处理 XML 的应用方式。

（1）JDOM。JDOM 是一个提供基于 Java 语言解决 XML 的开放源码项目，由 Brett McLaughlin 和 Jason Hunter 开发。目前，已经被接纳成为 JSP-102，是 Java 规范中的一个

成员。JDOM 本身没有解析器，它的一个特点是可以利用 DOM 或 SAX 来解析 XML，并提供相应的 API，多使用 Java 类而不是接口实现用 Java 处理 XML 数据。JDOM 的优势在于它可以很好的与 Java 程序结合，与 DOM 相比，更加适应 Java 语言开发 XML 应用。JDOM 的常见的包见表 14-1，常见的类见表 14-2。

表 14-1　JDOM 的常见的包

包	描　述
org. jdom	包含了所有的 XML 文档要素的 java 类
org. jdom. adapters	包含了与实现 DOM 接口的 java 类
org. jdom. filter	包含了 XML 文档的过滤器类
org. jdom. input	包含了创建 JDOM 文档的类
org. jdom. output	包含了输出 JDOM 文档的类
org. jdom. transform	基于 JAXP TRaX 类，包含了实现转换的类
org. jdom. xpath	包含支持 XPATH 的类

表 14-2　JDOM 的常见的类

类	描　述
org. jdom. Attribute	定义一个 XML 属性
org. jdom. Document	定义一个 XML 文档
org. jdom. Element	定义一个 XML 元素
org. jdom. Namespace	定义一个 XML 命名空间
org. jdom. Text	定义一个 XML 的文本内容
org. jdom. Content	父结点合法子内容 JDOM 对象的父类
org. jdom. transform. JDOMResult	表示 XSL 转换结果
org. jdom. transform. JDOMSource	表示要转换的数据源，可以是文档、元素或结点集
org. jdom. input. DOMBuilder	从已存在 org. w3c. dom. Document 创建一个 JDOM org. jdom. Document 文档
org. jdom. input. SAXBuilder	从文件、流、URL 等或 SAX 的 InputSource 实例用 SAX 解析创建的 JDOM 文档
org. jdom. input. SAXHandler	支持 SAXBuilder 的类
org. jdom. output. DOMOutputter	将 org. jdom. Document 作为 org. w3c. dom. Document 输出
org. jdom. output. Format	封装 DOMOutputter 的输出格式
SAXOutputter	作为 SAX 2. 0 事件流输出 JDOM 文档
XMLOutputter	作为 XML 字节流输出 JDOM 文档

（2）DOM4J。DOM4J 是一个开发源代码，是 JDOM 的一个智能分支，使用 Java 集合框架支持 DOM、SAX 和 JAXP，是集成处理 XML、XPATH 和 XSLT 的 Java 平台。是一个非常优秀的 Java XML API，可以处理大文档和流化文档。DOM4J 的常见的包见表 14-3，常见的类和接口见表 14-4。

JSP 可以使用上述 4 种解析 XML 的 API。但是在 JSP 直接使用 XML 数据存在一个问题，就是 Java 代码与 JSP 表示的数据无法分离，导致 JSP 程序的可读性差。在这样的情况下，可以通过 JavaBean 实现对 XML 文件的应用。

表 14-3 DOM4J 常见的包

包	说　　明
org. dom4j	提供定义 XML 文档对象模型的类和接口
org. dom4j. io	提供将 DOM4J 对象作为文本流输入输出 DOM 或 SAX 的类和接口
org. dom4j. util	提供 DOM4J API 的实用类
org. dom4j. rule	提供实现 XSLT 处理的类和接口
org. dom4j. bean	用于 JavaBean 检索和储存数据的类和接口
org. dom4j. datatype	提供支持 XML Schema 的类和接口
org. dom4j. dom	提供支持 W3C 对象模型的类和接口
org. dom4j. swing	集成 Swing 中的 TreeModel 和 TableModel 的适配器集合
org. dom4j. dtd	表示 DTD 的类和接口
org. dom4j. xpath	处理 XPATH 的类和接口

表 14-4 DOM4J 常见的类和接口

类/接口	说　　明
org. dom4j. Attribute	定义一个 XML 属性
org. dom4j. Document	定义一个 XML 文档
org. dom4j. Element	定义一个 XML 元素
org. dom4j. ElementHandler	定义一个 XML 元素处理器
org. dom4j. Node	定义一个结点
org. dom4j. Text	定义文本结点
org. dom4j. Xpath	定义一个 XPath 表达式
org. dom4j. io. DOMReader	定义浏览 DOM 树并创建 DOM4J 树
org. dom4j. io. DOMWriter	定义将 DOM4J 树作为 W3C 的 DOM 输出
org. dom4j. io. SAXReader	定义从 SAX 解析事件创建 DOM4J 树
org. dom4j. io. SAXWriter	定义将 DOM4J 树作为 SAX 内容处理器

14.1.2　JavaBean 处理 XML 数据

JavaBean 是可重用的组件，具有强大而灵活的功能。在 JSP 中，往往利用 JavaBean 封装对 XML 数据处理相关的业务逻辑。JSP 既可以用 JavaBean 动态生成 XML 文件，也可以通过 JavaBean 结合 JAXP API 实现对 XML 的解析，获取 XML 文件中的数据，执行相应的指令动作。JavaBean 有效地克服了 JSP 直接使用和处理 XML 文件中的不足，有效地体现了 JSP 页面和 Java 代码分离的原则，是一种常见处理和应用 XML 的技术。

JavaBean 处理 XML 数据，JSP 中可以通过<jsp:useBean>动作实现对 JavaBean 的应用。通过<jsp:setProperty>实现对 JavaBean 相关数据处理，以及用<jsp:getProperty>实现对相关数据的获取。

14.1.3　JSP 的自定义标签

JSP 的自定义标签是提供实现 JSP 页面与 Java 代码分离的又一处理方法。用户自定义标签允许用户自定义基于 XML 的标签。具体的步骤如下。

（1）编写处理标签的 Java 类。处理标签的 Java 类必须扩展 javax. servlet. jsp.

TagSupport 或 javax.servlet.jsp.BodyTagSupport 类。

javax.servlet.jsp.TagSupport 和 javax.servlet.jsp.BodyTagSupport 类的主要方法分别见表 14-5 和表 14-6。处理标签的 Java 类从本质上是编写处理 XML 标签的 JavaBean 组件类。所以,JSP 的自定义标签是一种特殊的 JavaBean。

表 14-5　javax.servlet.jsp.TagSupport 的方法

方　法	说　明
int doAfterBody()	默认处理标签主体
int doStartBody()	默认处理自定义标签的开始标志,返回 SKIP_BODY.
int doEndTag()	默认处理自定义标签的结束标志,返回 EVAL_PAGE.
Tag getParent()	返回封闭标签的上级标签
void setParent(Tag)	设置封闭标签的上级标签
void setPageContext(PageContext)	设置 PageContext 对象,在 doStartTag()调用之前调用
Object getValue(String)	获取参数对应的值
void setValue(String,Object)	设置名字-值对

表 14-6　javax.servlet.jsp.BodyTagSupport 的方法

方　法	说　明
int doAfterBody()	标签主体后执行
int doStartTag()	默认处理自定义标签的开始标志 ,返回 SKIP_BODY
int doEndTag()	默认处理自定义标签的结尾标志,返回 EVAL_PAGE
void doInitBody()	准备评估标签主体
BodyContent getBodyContent()	获取当前标签主体内容
void setBodyContent()	准备评估标签的实体
void release()	释放状态

(2) 创建描述标签库的文件。标签库描述文件是一种描述标签库的 XML 文件,但是它的文件扩展名必须为.tld。其中,描述标签库的文件中用特定含义的标签,具体内容见表 14-7。

表 14-7　描述标签库的元素

标　记	说　明	标　记	说　明
taglib	定义标签库	name	指定标签的名,或标签属性名
tlib-version	定义标签库的版本	tag-class	指定标签对应的处理类
jsp-version	指定 JSP 的版本	body-content	定义标签的主体内容
shortname	指定标签库的默认前缀	attribute	是 tag 的下级标签,定义标签元素的属性
uri	设置标签库的唯一访问标识符		
info	定义标签库的说明信息	required	定义属性是否必须
tag	定义标签	rtexprvalue	指定属性是否是 request-time 表示式

(3) 在 JSP 文件中应用标签。在 JSP 文件要应用自定义标签,首先要声明对标签库的引用。具体做法如下:

```
<%@taglib uri="标签库的唯一访问标识符" prefix="引用标签库的前缀名" %>
```

然后，在通过上述 taglib 指令 prefix 属性指定的标签库的前缀名来访问标签，具体形式如下：

```
<前缀名：标签名 [属性名="属性值"…]>
```

14.2　实验 14.1　JSP 生成 XML

实验目的：

（1）了解 JSP 直接生成 XML 的方法。
（2）了解利用 JavaBean 生成 XML 的方法。
（3）了解 JSP 生成 XML 的作用。

实验内容：

（1）在 JSP 文件中将给定的 XML 数据段直接生成 XML 文件。

（2）JSP 利用 JavaBean 生成 XML 文件。要求将某班级的“计算机基础”的成绩生成 XML 文件保留下来。

实验步骤：

练习 1：JSP 直接生成 XML 文件。

程序 medalist.jsp，**是可以输出 29 届奥运排名前 5 位的总奖牌榜**。见程序清单 14-1。要求：

（1）仔细阅读程序清单 14-1，要求修改该程序，使之能用浏览器运行可以正常输出的 XML 文件，运行结果类似图 14-1。

程序清单 14-1：

```
<!--medalist.jsp-->

<%@page contentType="text/xml" %>
<?xml version="1.0" encoding="UTF-8"?>

<medalist>
  <nation>
     <name>中国</name>
    <rank>1</rank>
    <total_medal>100</total_medal>
    <gold_medal>51</gold_medal>
     <silver_medal>21</silver_medal>
     <bronze_medal>28</bronze_medal>
  </nation>
```

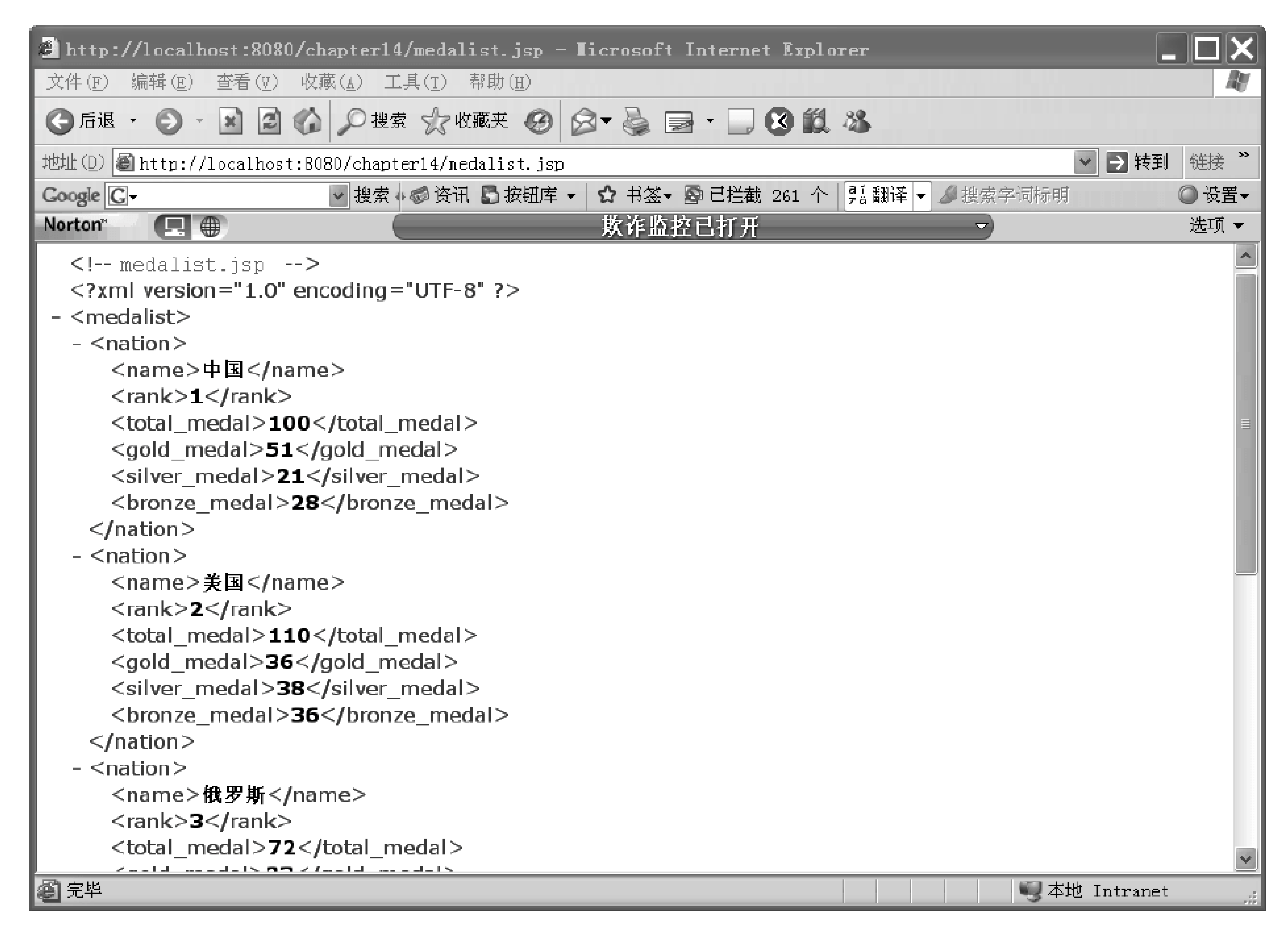


图 14-1　直接生成 XML 文件

```
<nation>
  <name>美国</name>
  <rank>2</rank>
  <total_medal>110</total_medal>
  <gold_medal>36</gold_medal>
  <silver_medal>38</silver_medal>
  <bronze_medal>36</bronze_medal>
</nation>

<nation>
  <name>俄罗斯</name>
  <rank>3</rank>
  <total_medal>72</total_medal>
  <gold_medal>23</gold_medal>
  <silver_medal>21</silver_medal>
  <bronze_medal>28</bronze_medal>
</nation>

<nation>
  <name>英国</name>
  <rank>4</rank>
  <total_medal>41</total_medal>
  <gold_medal>19</gold_medal>
  <silver_medal>13</silver_medal>
  <bronze_medal>15</bronze_medal>
```

```
  </nation>

  <nation>
    <name>德国</name>
   <rank>5</rank>
   <total_medal>41</total_medal>
   <gold_medal>16</gold_medal>
    <silver_medal>10</silver_medal>
    <bronze_medal>15</bronze_medal>
  </nation>
</medalist>
```

(2) 将上述程序清单 14-1 保存为 medalist2.jsp。要求：在 medalist2.jsp 中调用外部标签库实现 XSLT 的转换，已知为 medalist2.jsp 中的 XML 数据设计的 XSLT 文件为 medalist2.xslt，代码见程序清单 14-2。请修改后的 medalist2.jsp，将同样的 XML 数据直接生成为 XHTML MP 文件，使之输出结果如图 14-2。



图 14-2 生成 XHTML MP 类型文件

程序清单 14-2：

```
<?xml version="1.0" encoding="gb2312"?>
<!--medalist2.xslt-->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="xml" version="1.0" encoding="gb2312" indent="yes"/>

<xsl:template match="/">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
  <meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
  <title>奥运奖牌榜</title>
  </head>

<body>
     <table>
       <tr>
          <th>国家</th>
          <th>金</th>
          <th>银</th>
          <th>铜</th>
       </tr>
          <xsl:for-each select="medalist/nation" >
        <tr>
          <td><xsl:value-of select="name"/></td>
          <td><xsl:value-of select="gold_medal"/></td>
          <td><xsl:value-of select="silver_medal"/></td>
```

```
            <td><xsl:value-of select="bronze_medal"/></td>
         </tr>
         </xsl:for-each>
        </table>
    </body>
</html>
</xsl:template>

</xsl:stylesheet>
```

练习 2：利用 JavaBean 生成 XML 文件。

已知 MySQL 中有一数据库 StudentDB，该数据库的 computerTable 数据表（结构和数据见图 14-3）中保存 2007 级化学专业的“计算机基础”的总评成绩。已有一个 JavaBean 类 ScoreBean.java（代码见程序清单 14-3）和 ScoreHandleBean.java（代码见程序清单 14-4），其中 ScoreBean 保存单个学生总评成绩相关的方法，而 ScoreHandleBean 定义了处理从数据库 StudentDB 中获取和插入新记录的方法。要求根据提供的数据表和程序，编写 JSP 程序，将该班级的所有“计算机基础”的总评成绩生成 XML 文件，运行结果如图 14-4 所示。

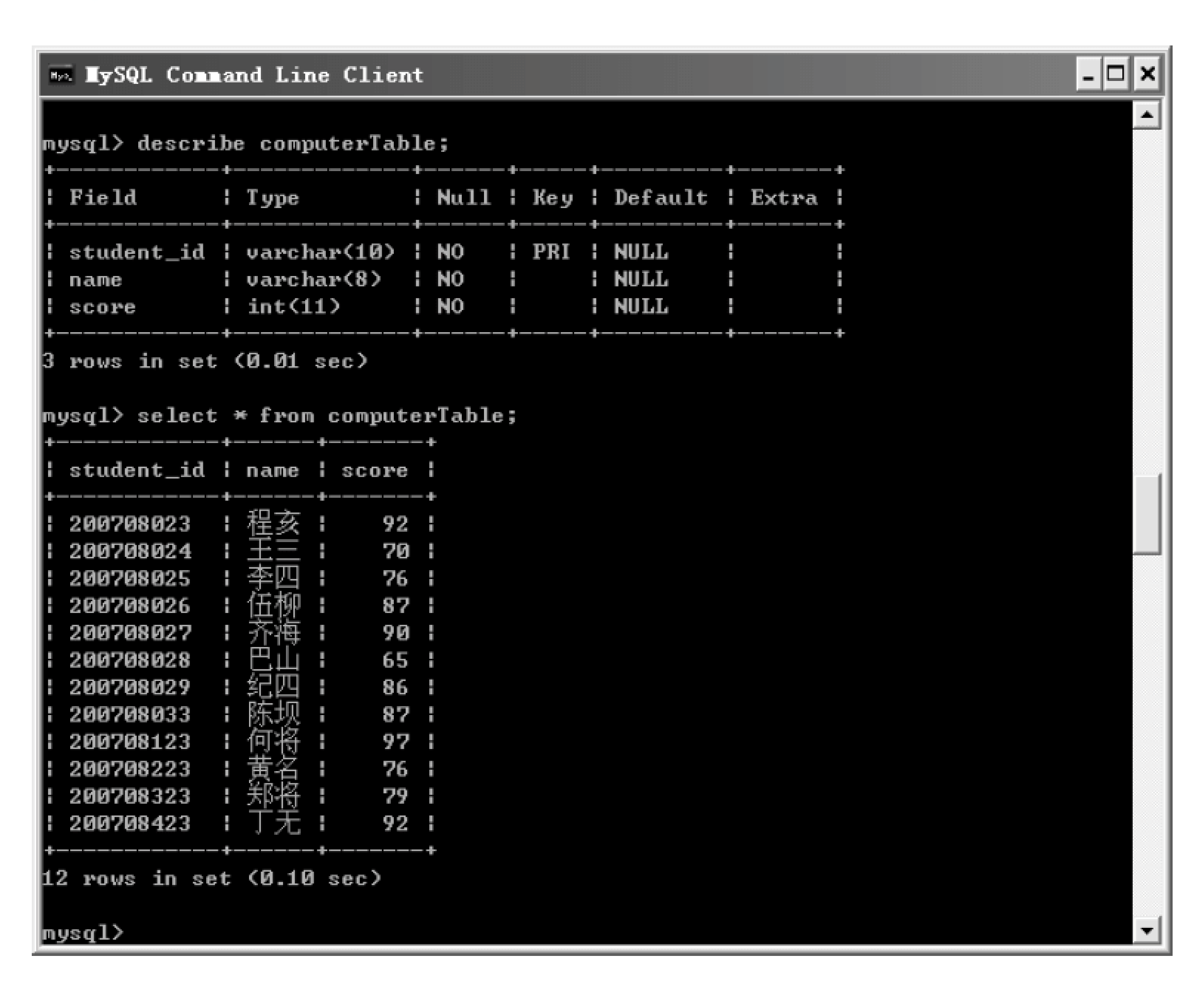


图 14-3　数据表 computerTable 的结构和数据

程序清单 14-3：

```
//ScoreBean.java
public class ScoreBean {
   String studentId;
   String studentName;
   int studentScore;

   public String getStudentId() {
```

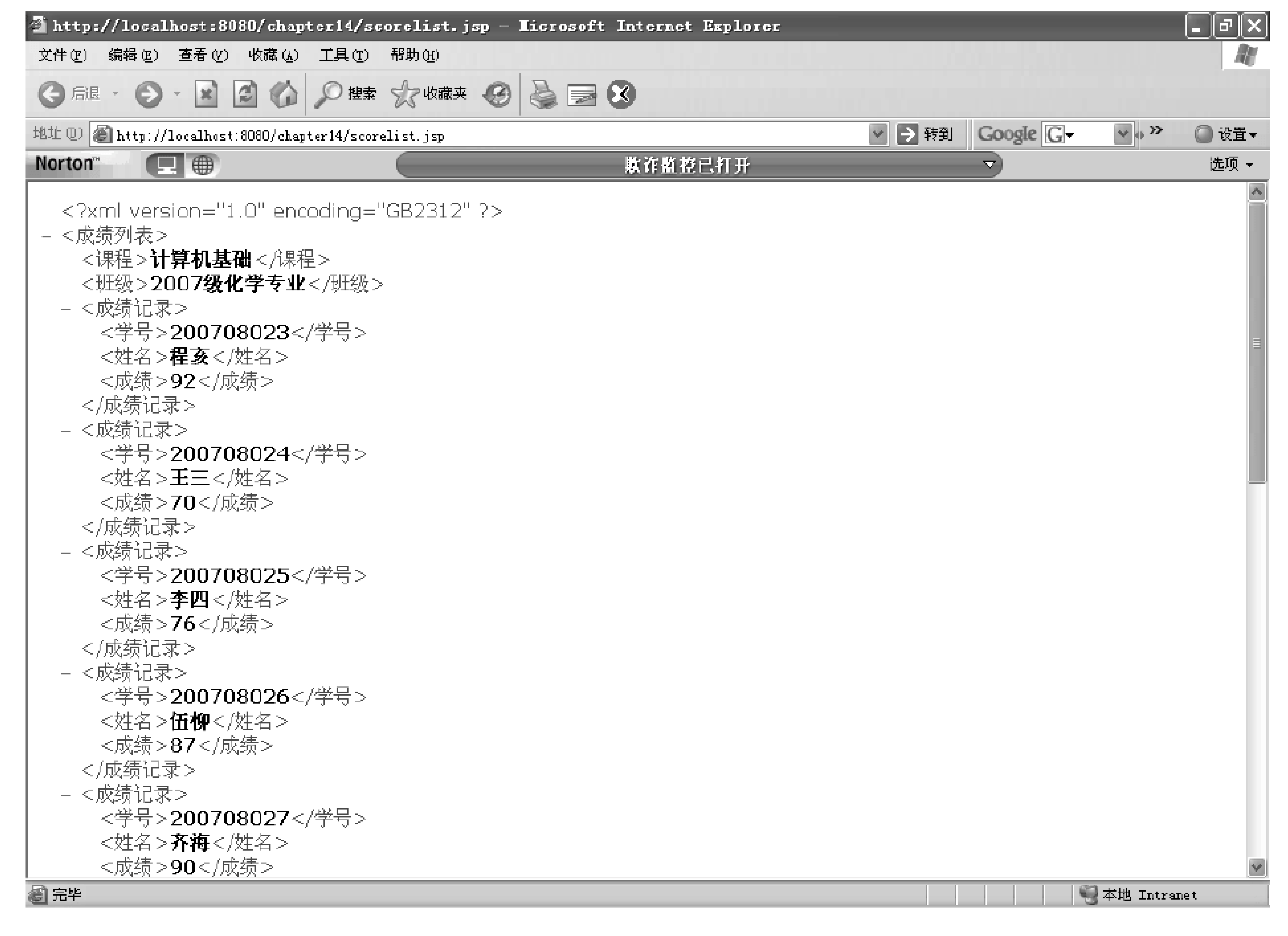


图 14-4 JavaBean 生成 XML 文件

```
//获取学号
  return studentId;
}

public void setStudentId(String studentId) {
//设置学号
  this.studentId=studentId;
}

public String getStudentName() {
//获取姓名
  return studentName;
}

public void setStudentName(String studentName) {
//设置姓名
  this.studentName=studentName;
}
public int getStudentScore() {
//获取成绩
  return studentScore;
}

public void setStudentScore(int studentScore) {
//设置姓名
```

```
    this.studentScore=studentScore;
  }
}
```

程序清单 14-4:

```
//ScoreHandleBean.java
import java.sql.*;
import java.util.*;
public class ScoreHandleBean {

  public ScoreBean[] getAllRecords() throws SQLException{
    ScoreBean record;
    Collection list=new ArrayList();
    try{
          Class.forName("com.mysql.jdbc.Driver").newInstance();
          Connection con=java.sql.DriverManager.getConnection(
                              "jdbc:mysql://localhost/StudentDB?useUnicode=true&"+
                              "characterEncoding=GB2312","root","123456");
          Statement stmt=con.createStatement();
          ResultSet rs=stmt.executeQuery("select * from computerTable");

          while(rs.next()){
           record=new ScoreBean();
           record.setStudentId(rs.getString(1));
           record.setStudentName(rs.getString(2));
           record.setStudentScore(rs.getInt(3));
           list.add(record);
           //将记录依次插入到列表中;
          }

          stmt.close();
          con.close();
     }catch(Exception e){
         e.printStackTrace();
     }
     ScoreBean[] records=(ScoreBean[])list.toArray(new ScoreBean[0]);
     //将列表中所有的元素转换成数组中的元素
     return records;
  }
}
```

14.3 实验 14.2 JSP 解析 XML

实验目的:

(1) 进一步了解 XML 的树状结构。

(2) 熟练掌握和运用JAXP API来解决实际问题。
(3) 了解DOM解析XML的原理和具体解析的过程。
(4) 了解SAX 2.0解析XML的原理和具体解析的过程。
(5) 比较DOM和SAX 2.0解析XML数据的异同,归纳二者的特点。

实验内容:

为某班设计一个学生管理简易系统,要求实现学生记录的插入、删除、修改以及对学生信息的查询浏览。所有的学生记录用XML文件保存,XML文件的验证见student.xsd。请分别用DOM API和SAX 2.0 API来实现同样的功能。并比较二者进行解析XML数据的特点。

实验步骤:

练习1:JSP用DOM API解析XML。

本练习的目的是了解JSP应用DOM来解析XML数据。本次练习是用DOM API来实现一个某班级学生基本情况简易管理系统,可以添加、删除、修改和浏览已知保存学生记录。已知保存学生记录的XML文件是student.xml,该文件的结构如student.xsd,代码见程序清单14-5。要求如下。

程序清单14-5:

```
<?xml version="1.0" encoding="UTF-8"?>
<!--student.xsd-->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="studentlist">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="student" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="student">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="id"/>
                <!--学号-->
                <xs:element ref="name"/>
                <!--姓名-->
                <xs:element ref="gender"/>
                <!--性别-->
                <xs:element ref="birthday"/>
                <!--出生日期-->
                <xs:element ref="email"/>
                <!--电子邮件-->
                <xs:element ref="address"/>
```

```
            <!--通信地址-->
            <xs:element ref="phone"/>
            <!--电话-->
            <xs:element ref="mobile-phone"/>
            <!--手机-->
        </xs:sequence>
    </xs:complexType>
</xs:element>

<xs:element name="phone">
<!--电话,(区号 4 位-号码 8 位)-->
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:pattern value="[0-9]{4}(-[0-9]{8})"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
<xs:element name="name">
<!--电话,类型为字符串-->
    <xs:simpleType>
        <xs:restriction base="xs:string"/>
    </xs:simpleType>
</xs:element>

<xs:element name="mobile-phone">
<!--手机,数字字符串,长度为 7 至 11 之间-->
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:pattern value="[0-9]+"/>
            <xs:minLength value="7"/>
            <xs:maxLength value="11"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>

<xs:element name="id">
<!--学号,字符串-->
    <xs:simpleType>
        <xs:restriction base="xs:string"/>
    </xs:simpleType>
</xs:element>

<xs:element name="gender">
<!--性别,只能取男或女-->
    <xs:simpleType>
```

```
            <xs:restriction base="xs:string">
                <xs:pattern  value="女|男"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:element>

    <xs:element name="email">
    <!--电子邮件,形式 xxx@xxx.xxx.xxx-->
        <xs:simpleType>
            <xs:restriction base="xs:string">
                <xs:pattern value="[^@]+@[^\.]+\..+"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:element>

    <xs:element name="birthday">
    <!--出生日期,日期型-->
        <xs:simpleType>
            <xs:restriction base="xs:date"/>
        </xs:simpleType>
    </xs:element>

    <xs:element name="address">
    <!--通信地址-->
        <xs:simpleType>
            <xs:restriction base="xs:string" />
        </xs:simpleType>
    </xs:element>
</xs:schema>
```

(1) 已知 StudentBean.java(代码见程序清单 14-6)和 StudentXMLBean.java(代码见程序清单 14-7)是 JavaBean 程序。其中,StudentBean.java 是记录一个学生的信息,而 StudentXMLBean.java 实现对 student.xml 文件的解析和相关处理,实现对 student.xml 插入记录、删除记录、修改记录和查询浏览记录。请仔细阅读程序清单 14-6 和程序清单 14-7,将程序清单 14-7 的 StudentXMLBean.java 中的 代码 1 ～ 代码 6 处补充完整。

程序清单 14-6:

```
//StudentBean.java
package student;

public class StudentBean {
private String id;
private String name;
private String gender;
private String birthday;
```

```
private String address;
private String email;
private String phone;
private String mobilePhone;

public String getId() {
//获取编号
    return id;
}

public void setId(String id) {
//设置编号
    this.id=id;
}

public String getName() {
//获取姓名
    return name;
}

public void setName(String name) {
//设置姓名
    this.name=name;
}

public String getGender() {
//获取性别
    return gender;
}

public void setGender(String gender) {
//设置性别
    this.gender=gender;
}

public String getBirthday() {
//获取出生日期
    return birthday;
}

public void setBirthday(String birthday) {
//设置出生日期
    this.birthday=birthday;
}
```

```
public String getAddress() {
//获取地址
    return address;
}

public void setAddress(String address) {
//设置地址
    this.address=address;
}

public String getEmail() {
//获取 e-mail
    return email;
}

public void setEmail(String email) {
//设置 e-mail
    this.email=email;
}

public String getPhone() {
//获取电话号码
    return phone;
}

public void setPhone(String phone) {
//设置电话号码
    this.phone=phone;
}

public String getMobilePhone() {
//获取手机号码
    return mobilePhone;
}

public void setMobilePhone(String mobilePhone) {
//设置手机号码
    this.mobilePhone=mobilePhone;
}
}
```

程序清单 14-7：

```
//StudentXMLBean.java
package student;
import org.w3c.dom.*;
```

```
import javax.xml.parsers.*;
import javax.xml.transform.*;
import javax.xml.transform.dom.*;
import javax.xml.transform.stream.*;
import javax.xml.xpath.*;
import java.io.*;

public class StudentXMLBean {
    DocumentBuilderFactory factory;
    DocumentBuilder builder;
    Document document;
    Element root;
    String filename="student.xml";
    public StudentXMLBean(){
        //默认构造方法
        try{
            factory= 代码 1 ;
            //创建文件制造工厂对象
            builder=factory.newDocumentBuilder();
            //创建文件制造器
            File file=new File(filename);
            document=builder. 代码 2 ;
            //创建文档对象
            root=document.getDocumentElement();
        }catch(Exception e){
            e.printStackTrace();
        }
    }

    public void saveXML(){
        //将发生变化 XML 数据保存在 XML 文件中
        try{
            TransformerFactory transFactory= 代码 3 ;
            //创建转换工厂对象
            Transformer transformer=transFactory.newTransformer();
            DOMSource source=new DOMSource(document);
            StreamResult result=new StreamResult(new File(filename));
            transformer. 代码 4 ;
            //实现转换
        }catch(Exception e){
            e.printStackTrace();
        }
    }
```

```
public void insertAElement(Element student,String tagName,String content){
    //插入一个元素;
    Node element=document.createElement(tagName);
    element.setTextContent(content);
    student.appendChild(element);
}

public void setAElement(Element element,String tagName,String content){
    //设置指定元素 element 中,tagName 子元素的值为 content;
    element.getElementsByTagName(tagName).item(0).setTextContent(content);
}

public String getAElementValue(Element element,String tagName){
    //获取指定元素 element 中,tagName 子元素的值
    return element.getElementsByTagName(tagName).item(0).getTextContent();
}

public Element selectNode(String id,Object source){
    //选择指定编号为 id 的结点
    Element result=null;
    XPathFactory xpathFactory=XPathFactory.newInstance();
    XPath xpath=xpathFactory.newXPath();
    try {
        result= (Element)xpath.evaluate("代码 5",source, XPathConstants.NODE);
        //检索指定 id 编号的所有元素
    } catch (XPathExpressionException e) {
        e.printStackTrace();
    }
    return result;
}

public NodeList selectNodes(Object source){
    //选择 studentlist/student 元素的所有结点
    NodeList result=null;
    XPathFactory xpathFactory=XPathFactory.newInstance();
    XPath xpath=xpathFactory.newXPath();
    try {
        result= (NodeList)xpath.evaluate("/studentlist/student", source,
        XPathConstants.NODESET);
    } catch (XPathExpressionException e) {
      System.out.println(e.getMessage());
        e.printStackTrace();
    }
    return result;
}
```

```java
public void insertARecord(StudentBean record){
    //将 record 的信息插入 XML 文件中
    Element student;
    student=document.createElement("student");

    insertAElement(student,"id",record.getId());
    insertAElement(student,"name",record.getName());
    insertAElement(student,"gender",record.getGender());
    insertAElement(student,"birthday",record.getBirthday());
    insertAElement(student,"address",record.getAddress());
    insertAElement(student,"email",record.getEmail());
    insertAElement(student,"phone",record.getPhone());
    insertAElement(student,"mobile-phone",record.getMobilePhone());

    root.代码 6;
    //插入新元素到根元素中
    saveXML();
}

public boolean deleteARecord(String id){
    //删除学号为 id 的记录,如果删除成功返回真,否则返回假
    Element element=selectNode(id,root);
    try{
        element.getParentNode().removeChild(element);
        saveXML();
    }catch(Exception e){
        return false;
    }
    return true;
}

public boolean updateARecord(String id,StudentBean record){
    //修改学号为 id 的记录,如果修改成功返回真,否则返回假
    Element element=selectNode(id,root);
    try{
        setAElement(element,"id",record.getId());
        setAElement(element,"name",record.getName());
        setAElement(element,"gender",record.getGender());
        setAElement(element,"birthday",record.getBirthday());
        setAElement(element,"email",record.getEmail());
        setAElement(element,"address",record.getAddress());
        setAElement(element,"phone",record.getPhone());
        setAElement(element,"mobile-phone",record.getMobilePhone());
    }catch(Exception e){
```

```
            return false;
        }
        saveXML();
        return true;
    }

    public StudentBean queryARecord(String id){
        //查询学号为 id 的记录,如果查询成功返回保存查询信息的记录
        Element element=selectNode(id,root);
        StudentBean record=new StudentBean();
        record.setId(getAElementValue(element,"id"));
        record.setName(getAElementValue(element,"name"));
        record.setGender(getAElementValue(element,"gender"));
        record.setBirthday(getAElementValue(element,"birthday"));
        record.setEmail(getAElementValue(element,"email"));
        record.setAddress(getAElementValue(element,"address"));
        record.setPhone(getAElementValue(element,"phone"));
        record.setMobilePhone(getAElementValue(element,"mobile-phone"));
        return record;
    }

    public StudentBean[] getAllRecords(){
        //获取 student.xml 文件中的所有记录
        NodeList list=selectNodes(root);
        int length=list.getLength();
        if(length<1) return null;
        StudentBean[] result=new StudentBean[length];
        for(int i=0;i<list.getLength();i++){
            result[i]=new StudentBean();
            Node node=list.item(i).getFirstChild();
            while(node!=null){
                if(node.getNodeType()==Node.ELEMENT_NODE){
                    if(node.getNodeName().equals("id")){
                        result[i].setId(node.getTextContent());
                    }
                    else if(node.getNodeName().equals("name")){
                        result[i].setName(node.getTextContent());
                    }
                    else if(node.getNodeName().equals("gender")){
                        result[i].setGender(node.getTextContent());
                    }
                    else if(node.getNodeName().equals("birthday")){
                        result[i].setBirthday(node.getTextContent());
                    }
```

```
                    else if(node.getNodeName().equals("email")){
                        result[i].setEmail(node.getTextContent());
                    }
                    else if(node.getNodeName().equals("address")){
                        result[i].setAddress(node.getTextContent());
                    }
                    else if(node.getNodeName().equals("phone")){
                        result[i].setPhone(node.getTextContent());
                    }
                    else if(node.getNodeName().equals("mobile-phone")){
                        result[i].setMobilePhone(node.getTextContent());
                    }
                }
                node=node.getNextSibling();//获取下一个兄弟结点;
            }//end while
        }//end for
        return result;
    }
}
```

(2) 编写相关的 JSP 文件,利用上述 JavaBean 组件,设计并实现学生基本信息简易管理系统的页面,并实现对学生记录的插入、删除、修改以及查询浏览记录的具体操作。该系统的首页类似图 14-5。



编号	姓名	性别	出生日期	通信地址	电子邮件	电话	手机
20070903431	王方	女	1992-01-23	XXXX省XXXX市北京路2323号江海花园12栋2单元1001室	wangfang12@yahoo.com.cn	0321-32432434	13088888888
20070903442	王真	男	1991-04-21	XXXX省XXXX市杭州路211号111信箱	wangzhen11@yahoo.com.cn	0321-23232223	13077777777
20070900002	王方湖	女	1992-11-23	XXXX省XXXX市北京路2323号江海花园12栋3单元1001室	wangfang123@yahoo.com.cn	0321-52432434	13066888888
20070903443	王真将	男	1991-04-23	XXXX省XXXX市杭州路211号111信箱	wangzhen1222@yahoo.com.cn	0321-23232223	13077777777
20070903434	刘方化	女	1993-01-23	XXXX省XXXX市北京路2323号江海花园12栋2单元1001室	wangfang322@yahoo.com.cn	0321-32432434	13088888888
20070903445	王佳	男	1991-04-21	XXXX省XXXX市杭州路211号111信箱	wangzhen124@yahoo.com.cn	0321-23232223	13077777777
20070903888	王嘉	女	1992-01-23	XXXX省XXXX市北京路2323号江海花园12栋2单元1001室	wangfang121@yahoo.com.cn	0321-32432434	13088888888
20070903446	王明	男	1991-04-21	XXXX省XXXX市杭州路211号111信箱	wangzhen34@yahoo.com.cn	0321-23232223	13077777777
20070903432	王方芳	女	1992-01-23	XXXX省XXXX市北京路2323号江海花园12栋2单元1001室	wangfang56@yahoo.com.cn	0321-32432434	13088888888
20070903444	王真南	男	1991-04-21	XXXX省XXXX市杭州路211号111信箱	wangzhen@yahoo.com.cn	0321-23232223	13077777777

图 14-5 管理系统的首页

练习 2：JSP 应用 SAX 2.0。

本练习的目的是了解 JSP 应用 SAX 2.0 解析 XML 数据。本次练习是用 SAX 2.0 API 来实现一个某班级学生基本情况简易管理系统，可以实现浏览已知保存学生记录。

练习 3：JSP 应用 DOM4J 解析 XML。

这是一个扩展练习。具体要求是从网站 http://www.dom4j.org 下载 DOM4J API 以及相关的文档，了解和学习使用 DOM4J API 并利用 DOM4J API 实现练习 14.1 中所有的功能。

练习 4：JSP 应用 JDOM 解析 XML。

这是一个扩展练习。具体要求是从网站 http://www.jdom.org 下载 JDOM API 以及相关的文档，了解和学习使用 JDOM 并利用 JDOM 实现练习 14.1 中所有的功能。

思考：以上 4 种解析 XML 的方式的特点各是什么？

14.4 实验 14.3 JSP 自定义标签

实验目的：

（1）进一步了解 JSP 应用 XML。

（2）了解 JSP 自定义标签的原理。

（3）了解 JSP 使用自定义标签的过程。

实验内容：

本次实验由两个练习构成。

（1）自定义一个 hello 标签，该标签 value 属性取值为字符串。该标签的作用是输出"value 取值，欢迎学习 Web 开发技术实用教程"字符串。

（2）将第 12 章的实验 12.2 的简易论坛重新用 JSP+JavaBean+Servlet+XML 重新开发。要求在 JSP 页面中全部使用标签，不允许使用 Java 代码。

实验步骤：

练习 1：自定义 hello 标签。

已知 hello.jsp，具体代码见程序清单 14-8。该 JSP 页面使用了自定义标签 hello，来实现对限定内容的字符串信息的输出。运行结果类似图 14-6。具体要求如下。

程序清单 14-8：

```
<!--hello.jsp-->
<%@page contentType="text/html; charset=gb2312" language="java"%>
<%@taglib uri="/WEB-INF/HelloTagLib.tld" prefix="m"%>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>使用自定义标签</title>
```

```
</head>

<body>
<p>
带属性的标签输出：<br/>
<h3><m:hello value="江山"  /></h3><br/>
不带属性的标签输出：<br/>
<h3><m:hello  /></h3>
</p>
</body>
</html>
```



图 14-6 hello.jsp 的运行结果

（1）HelloTag.java 是一个处理实现输出字符串的 hello 标签的处理类，代码见程序清单 14-9。请将程序清单 14-9 中 代码 1 和 代码 2 处补充完整。

程序清单 14-9：

```
//HelloTag.java
package tags;
import java.io.*;
import javax.servlet.jsp.tagext.*;
import javax.servlet.jsp.*;

public class HelloTag extends TagSupport {
  private String value;
//设置标签属性 value
  public int doStartTag() throws JspException{
    try{
        if(value==null){
            setValue("游客");
        }
     代码 1 println(getValue()+",学习《Web 开发技术实用教程》.");
```

```
        //输出字符串
        }catch(Exception e){
            throw new JspException(e.getMessage());
        }
        return 代码 2;
        //返回忽略标签主体
    }

    public void setValue(String value){
    //设置属性 value
        this.value=value;
    }

    public String getValue(){
    //获取属性 value
        return this.value;
    }
}
```

（2）下列程序清单 14-10 是创建标签库的描述文件 HelloTagLib.tld，请将程序清单 14-10 中 代码 3 和 代码 4 处补充完整，实现对 hello 标签的定义。并将 HelloTagLib.tld 保存在对应 web 应用的 WEB-INF 目录下。

程序清单 14-10：

```
<?xml version="1.0" encoding="gb2312"?>
<!--HelloTagLib.tld-->
<taglib>
  <tlib-version>1.0</tlib-version>
  <jsp-version>1.2</jsp-version>
  <shortname>hello</shortname>
  <uri> 代码 3 </uri>
  <!--设置标签库的唯一访问标识符 -->
  <info>自定义标签 hell</info>
  <tag>
    <name>hello</name>
    <tag-class> 代码 4 </tag-class>
    <!--指定标签的处理类 HelloTag -->
    <body-content>empty</body-content>
    <info>输出字符串</info>
    <attribute>
      <name>value</name>
      <required>false</required>
      <rtexprvalue>true</rtexprvalue>
    </attribute>
```

```
  </tag>
</taglib>
```

（3）请编写对应应用的 web. xml，在 web. xml 中部署自定义标签库 HelloTagLib. tld，运行 hello. jsp 页面可以实现对标签 hello 的应用。观察运行结果。

练习 2：JSP＋XML 开发一个论坛。

模仿 ForumLite 2.1 的结构设计一个具有最基本功能的简易论坛。此系统要实现允许用户浏览论坛、发表新论点，也可以浏览并回复论坛的已有论点。论坛中所有的数据信息用 XML 保存。要求自定义标签，实现对 XML 数据的访问、修改、插入、删除等功能。具体功能要求：

（1）任何用户可以直接发表论题，论题包括标题、内容、发表者的电子邮件以及发表日期；如果有用户对该论题回复，要求能显示已回复论题的次数。

（2）任何用户可以浏览已发表的论题，并可以针对特定的论题进行回复。回复信息包括标题、内容、回复者的电子邮件以及回复的时间。

（3）要求所有论题以分页的形式显示，每页面 10 个论题。

（4）要求某论题的所有回复也以分页显示，每页面 10 个回复。

思考：比较 JSP＋XML 开发论坛与 JSP＋MySQL 开发论坛的异同。请回答结合 JSP＋XML＋MySQL开发论坛，XML 和 MySQL 的各起什么作用效果会更好。

高等学校计算机专业教材精选

计算机技术及应用

信息系统设计与应用(第 2 版)	赵乃真	即将出版

计算机硬件

单片机与嵌入式系统开发方法	邵贝贝	即将出版
基于 ARM 嵌入式 μCLinux 系统原理及应用	李岩	ISBN 978-7-302-18693-9

计算机基础

计算机科学导论教程	黄思曾	ISBN 978-7-302-15234-7
计算机应用基础教程(第 2 版)	刘旸	ISBN 978-7-302-15604-8

计算机原理

操作系统原理教程(第 2 版)	孟静	即将出版
计算机系统结构	李文兵	ISBN 978-7-302-17126-3
计算机组成原理(第三版)	李文兵	ISBN 978-7-302-13546-3
微型计算机操作系统基础——基于 Linux/i386	任哲	ISBN 978-7-302-17800-2
微型计算机原理与接口技术应用	陈光军	ISBN 978-7-302-16940-6

软件工程

软件工程实用教程	范立南	即将出版

数理基础

离散数学及其应用	周忠荣	ISBN 978-7-302-16574-3

算法与程序设计

C++ 程序设计	赵清杰	ISBN 978-7-302-18297-9
C++ 程序设计实验指导与题解	胡思康	ISBN 978-7-302-18646-5
C 语言程序设计教程	覃俊	ISBN 978-7-302-16903-1
C 语言上机实践指导与水平测试	刘恩海	ISBN 978-7-302-15734-2
Java 程序设计(第 2 版)	娄不夜	即将出版
Java 程序设计教程	孙燮华	ISBN 978-7-302-16104-2
Java 程序设计实验与习题解答	孙燮华	ISBN 978-7-302-16411-1
Visual Basic 上机实践指导与水平测试	郭迎春	ISBN 978-7-302-15199-9
程序设计基础习题集	张长海	ISBN 978-7-302-17325-0
计算机程序设计经典题解	杨克昌	ISBN 978-7-302-16358-9
数据结构	冯俊	ISBN 978-7-302-15603-1

数据库

SQL Server 2005 实用教程	范立南	ISBN 978-7-302-20260-8
数据库原理与应用案例教程	郑玲利	ISBN 978-7-302-17700-5

图形图像与多媒体技术

AutoCAD 2008 中文版机械设计标准实例教程	蒋晓	ISBN 978-7-302-16941-3
Pro/ENGINEER 标准教程	樊旭平	ISBN 978-7-302-18718-9
Photoshop(CS2 中文版)标准教程	施华锋	ISBN 978-7-302-18716-5
计算机图形学基础教程(Visual C++ 版)	孔令德	ISBN 978-7-302-17082-2
计算机图形学实践教程(Visual C++ 版)	孔令德	ISBN 978-7-302-17148-5
计算机图形学基础教程(Visual C++ 版)习题解答与编程实践	孔令德	即将出版

网络与通信技术

Web 开发技术实用教程	陈轶	ISBN 978-7-302-17435-6
Web 开发技术实验指导	陈轶	ISBN 978-7-302-19942-7
Web 数据库编程与应用	魏善沛	ISBN 978-7-302-17398-4
Web 数据库系统开发教程	文振焜	ISBN 978-7-302-15759-5
实用网络工程技术	王建平	ISBN 978-7-302-20169-4
计算机网络技术与实验	王建平	ISBN 978-7-302-15214-9
计算机网络原理与通信技术	陈善广	ISBN 978-7-302-15173-9
网络安全基础教程	许伟	ISBN 978-7-302-19312-8

读者意见反馈

亲爱的读者：

感谢您一直以来对清华版计算机教材的支持和爱护。为了今后为您提供更优秀的教材，请您抽出宝贵的时间来填写下面的意见反馈表，以便我们更好地对本教材做进一步改进。同时如果您在使用本教材的过程中遇到了什么问题，或者有什么好的建议，也请您来信告诉我们。

地址：北京市海淀区双清路学研大厦 A 座 602　计算机与信息分社营销室 收
邮编：100084　电子邮件：jsjjc@tup.tsinghua.edu.cn
电话：010-62770175-4608/4409　邮购电话：010-62786544

教材名称：Web 开发技术实验指导

ISBN：978-7-302-19942-7

个人资料

姓名：________ 年龄：____ 所在院校/专业：__________

文化程度：______ 通信地址：______________

联系电话：______ 电子信箱：______________

您使用本书是作为：□指定教材 □选用教材 □辅导教材 □自学教材

您对本书封面设计的满意度：

□很满意 □满意 □一般 □不满意　改进建议__________

您对本书印刷质量的满意度：

□很满意 □满意 □一般 □不满意　改进建议__________

您对本书的总体满意度：

从语言质量角度看 □很满意 □满意 □一般 □不满意

从科技含量角度看 □很满意 □满意 □一般 □不满意

本书最令您满意的是：

□指导明确 □内容充实 □讲解详尽 □实例丰富

您认为本书在哪些地方应进行修改？（可附页）

____

____

您希望本书在哪些方面进行改进？（可附页）

____

____

电子教案支持

敬爱的教师：

为了配合本课程的教学需要，本教材配有配套的电子教案（素材），有需求的教师可以与我们联系，我们将向使用本教材进行教学的教师免费赠送电子教案（素材），希望有助于教学活动的开展。相关信息请拨打电话 010-62776969 或发送电子邮件至 jsjjc@tup.tsinghua.edu.cn 咨询，也可以到清华大学出版社主页（http://www.tup.com.cn 或 http://www.tup.tsinghua.edu.cn）上查询。